FARGO_THORIN 1.0 User Guide

Ondřej Chrenko

Charles University in Prague, Institute of Astronomy

July 6, 2017

Contents

Preface

Thank you for your interest in using FARGO_THORIN. The following sections will introduce the program, its features and they will also provide some hints on how to use the code for scientific simulations of protoplanetary systems.

FARGO_THORIN is a free program and still under development. I would be very grateful for any feedback from the users. Do not hesitate to contact me if you have a question regarding the code, or a suggestion for an improvement, if you encounter a bug, or if you would like to contribute to the upcoming versions. I would also be pleased to participate in research collaborations related to applications of FARGO_THORIN in planetary sciences.

Ondřej Chrenko
chrenko@sirrah.troja.mff.cuni.cz
`http://sirrah.troja.mff.cuni.cz/~chrenko/`

Chapter 1

Introduction

1.1 What is FARGO_THORIN?

FARGO_THORIN – or THORIN for short – stands for FARGO with Two-fluid HydrOdynamics, the Rebound integrator Interface and Non-isothermal gas physics. The code was introduced in Chrenko et al. (2017) ('Eccentricity excitation and merging of planetary embryos heated by pebble accretion') and its primary purpose is to study protplanetary systems, specifically mutual interactions between a gaseous disk, a disk of small solid particles (pebbles) and embedded protoplanets.

THORIN is built on top of the 2D FARGO hydrocode (Masset 2000), several algorithms are adopted from the FARGO-ADSG version (Baruteau & Masset 2008) and the code is interfaced with the REBOUND integrator package (Rein & Liu 2012).

1.2 Where to start & what to know

The program is distributed as an archive containing the following:

- `/in_relax` – directory containing the setup files of the first example simulation.
- `/in_wplanet` – directory containing the setup files of the second example simulation.
- `/src_main` – directory containing the source files of the THORIN code.
- `/src_reb` – directory containing the source files of the REBOUND integrator package to be linked to THORIN as a shared library.
- `GNUGPL3` – a copy of the GNU General Public License, 3rd version.
- `LICENSE` – license of the distribution.
- `README` – readme file of the distribution (explains the basic compilation and how to run the examples).

- `UserGuide.pdf` – this file.
- `refman.pdf` – developer's guide generated by DOXYGEN.

Before reading any further and before using the code, please read the license agreement. After that, it might be a good idea to check the readme file. It will guide you quickly through the program compilation and it will help you run your first example simulation using THORIN. You can return to this manual afterwards to learn more about how to setup your own simulation.

This guide assumes the user to have at least the basic knowledge and experience with the standard 2D FARGO hydrocode. You may benefit from reading through the homepage of the FARGO project (`http://fargo.in2p3.fr/-Legacy-archive-`) because many features of THORIN are quite similar.

Although THORIN is interfaced with the REBOUND integrator package, there is no need for you to know REBOUND, the interface will do the hard work for you. However, the REBOUND package is quite extended and only a handful of its possibilities is actually used in THORIN. In case you would like to explore the various aspects of REBOUND, see the website of the project (`http://rebound.readthedocs.io/en/latest/`).

For those who would like to modify THORIN, this guide will probably be too brief. You may find the developer's guide to be a helpful source of information, although it is merely an overview of the source code. The guide is generated by DOXYGEN, it is distributed with this archive and it is also available online
(access it from `http://sirrah.troja.mff.cuni.cz/~chrenko`) in a more compact form. In the end, you will need to browse through the source code itself. I tried to label all the important differences and additions with respect to FARGO with a standard C-like comment `/* */` containing '#THORIN'. In case you are familiar with the FARGO code, these labels might help you find out what is new. The labels are not used in source files (`*.c`) which are brand new in THORIN. These are especially the files `EnergySources.c`, `Pebbles.c` and `ReboundInterface.c` You will also notice that files named `Force.c` and `Pframeforce.c` were almost completely rewritten.

1.3 Implemented physics

Let us summarize the new physical modules of THORIN. Please note that a detailed description of the implemented physics is provided in Chrenko et al. (2017).

- **Non-isothermal gas disk** with implicit solution of the energy equation. The implemented energy source terms are: Compressional heating, viscous heating, stellar irradiation, vertical escape of radiation, radiative diffusion in the midplane and radiative feedback to accretion heating of embryos.
- **Planets evolved in 3D**, with close encounters allowed. The orbital evolution due to mutual planetary interactions is calculated using a high-accuracy integration technique with adaptive time step sub-division and detection of collisions. Collisions

are resolved as planetary mergers. Inclination damping is provided by an artificial vertical force.

- **Refined treatment of the planet-disk gravitational interaction.** Vertical averaging of the gravitational potential leads to relatively precise migration rates and the potential smoothing length is reduced from fractions of the local pressure scale height to fractions of the Hill sphere radius, making the potential well deeper than in the standard FARGO simulations. This is especially suitable for low-mass planets with their Hill sphere being smaller than the pressure scale height.

- **Pebble disk** represented by a single Eulerian, pressureless and inviscid fluid. The pebble dynamics is affected by the Epstein gas drag and optionally by the diffusive effects. We also implemented the drag back-reaction term into the Navier-Stokes equation for the gas. Pebbles, if incorporated into simulations, are automatically being accreted by planets. Their Stokes numbers, necessary to properly impose the drag force and calculate the accretion rates, are inferred from a steady-state model of a pebble disk in the coagulation-drift equilibrium, like in Lambrechts & Johansen (2014). At each radius, such a disk is dominated by a single pebble size, allowing to reasonably represent the pebble disk by a single fluid. Numerical solution of the pebble fluid motion equation follows a semi-implicit numerical scheme and the transport step utilises the FARGO algorithm.

1.4 Technical features

Like FARGO, THORIN is entirely written in C. It also supports multithreading on shared-memory systems using OpenMP and moreover, it supports parallelism on distributed-memory systems using MPI. The MPI implementation follows the radial domain splitting of the original FARGO code.

It is possible to compile THORIN in four different ways to build a sequential, multi-threaded, MPI-parallel or combined (MPI with OpenMP) executable. A possible way of producing these builds is given by the shell scripts placed in `/src_main`. These are named `make.sh`, `make_omp.sh`, `make_para.sh` and `make_para_omp.sh`, respectively. For most of our numerically demanding simulations, running the code with MPI proved itself to be the best option.

Regarding the OpenMP implementation, we hard-coded into THORIN a couple of omp functions which affect how the computation is performed. First, an automatic shutout of dynamic thread management is triggered. Second, the number of threads is set fixed to the total number of available cores detected on the machine. This behaviour can be sometimes undesirable when one wants to use a specific number of cores. To switch it off, open `/src_main/main.c`, find the first conditional block `#ifdef OPENMP ... #endif` and comment it out. After that, you can recompile and specify the number of threads by setting the environment variable `OMP_NUM_THREADS`.

If you use OpenMP on a machine with hyper-threading support, you might experience very small (or none) acceleration. In such a case, try setting the environment variables `OMP_PROC_BIND = true` and/or `OMP_PLACES = cores` and/or `OMP_DYNAMIC = false`.

We emphasize that the OpenMP implementation contains one major drawback – in the source file `/src_main/Pframeforce.c`, we had to use the `atomic` omp pragmas at four lines of the code. These pragmas reduce the efficiency of the OpenMP implementation and we plan to redesign the algorithm in future so they can be removed.

Combination of MPI with OpenMP is possible, but not thoroughly tested. Please note that each node will try to perform the steps indicated above for its own OpenMP sub-calculation (i.e. it will disable dynamic thread management and set the number or threads to the number of cores).

Chapter 2

Input and output

2.1 How to start a simulation

To start a simulation run, you can proceed exactly as in FARGO, be executing:

(MPI only →) `mpiexec` MPISET (← MPI only) `./thorin` –SWITCH PARAM
where you have to replace 'MPISET' with valid settings of the `mpiexec` command, 'SWITCH' should be replaced with command line switches (or can be omitted together with –) and 'PARAM' should contain the path to an input parametric file.

See also the readme file for a few examples of how to start THORIN.

2.2 Input files

Via the command above, you provide an input parametric file to be read by THORIN. The parametric file controls the entire simulation settings. It must be formatted as in standard FARGO: Lines starting with a hash will be omitted; give one parameter per line by writing its full name (not case sensitive), followed by an appropriate value to be assigned, (optionally) followed by a comment.

Some of the parameter names are the same as in FARGO but there is also a set of new parameters. They are all introduced later in Sect. 2.4. The allowed parameter values depend on the parameter type which can be integer, real or string. Some string parameters act like booleans (you set `YES` or `NO`), or contain the algorithm name (e.g. the `Transport` parameter can have value `FARGO`) or contain a path to a file (e.g. the `PlanetConfig` parameter). The allowed values of parameters will be also clarified in Sect. 2.4.

Along with the parametric file, you must provide a configuration file of the planetary system, accessible by the path from the `PlanetConfig` parameter. The algorithm always assumes there is at least one planet in the calculation thus you must also give at least one entry in the planet configuration file. In case you do not want the disk to be perturbed by the planet significantly, you can assign some tiny mass to the planet.

The planet configuration file may contain comments formated as lines starting with a hash. Entry for a planet should be given on one line, starting with the planet's name (e.g. 'Jupiter'), followed by the planet mass in Solar masses, followed by a set of six initial Keplerian orbital elements: The semimajor axis in AU, eccentricity ranging from 0 to < 1, inclination, longitude of ascending node, longitude of pericenter, and true anomaly, all angles being given in degrees.

There are a few optional input files related to the `InitializeFromFile` parameter, as will be explained in Sect. 2.4.

2.3 Command line switches

All the command line switches in THORIN were inherited from FARGO but only a handful of them was actually used and tested. These include

- `–m` switch which tells the code to merge the outputs produced by different CPUs in an MPI run.
- `–s number` switch which tells the code to restart from outputs numbered with `number`.
- `–t` switch which tells the code to provide some CPU consumption information. We find this useful only for a sequential build.
- `–v` switch which outputs some information about the simulation settings. But we emphasize here that part of the information in not valid as the corresponding output was not adjusted to modifications in THORIN. We plan to improve this in future versions.

Any other command line switches from FARGO were not tested in THORIN and although some of them might still work, we advise to use them carefully.

2.4 Simulation setup

Standard parameters. The following parameters have the same effect as in the standard FARGO code:

`DT`, `Sigma0`, `Ninterm`, `Ntot`, `OutputDir`, `Transport` (we always set this to `FARGO`), `PlanetConfig` (please note that the configuration file itself should have a different structure compared to standard FARGO), `MassTaper`, `RadialSpacing` (we only tested `ARITHMETIC`) , `Nrad`, `Nsec`, `Rmin`, `Rmax`, `Viscosity`, `AlphaViscosity`, `SigmaSlope`, `OmegaFrame`, `Disk` (we always set this to `YES`) ,`Frame`, `WriteDensity`, `WriteVelocity`.

Modified standard parameters. The following parameters are inherited from the standard FARGO code but allow for a new option or have a different effect:

- `InnerBoundary` has a new option named `DAMPING`, following the condition from de Val-Borro et al. (2006). See Sect. 2.7 in Chrenko et al. (2017) for a description.

We emphasize that only `RIGID` and `DAMPING` boundaries were thoroughly tested in simulations. The `DAMPING` condition has two sub-options controlled by the `DampTowards` parameter.

- `ThicknessSmoothing` controls the smoothing length in a new gravitational potential which was implemented according to Klahr & Kley (2006), see also Eq. (37) in Chrenko et al. (2017). We emphasize this is the only type of smoothing allowed in THORIN.

- `RocheSmoothing` parameter cannot be used in THORIN. Although the parameter was retained for possible future extensions, it has no effect in the current version of the code.

- `AspectRatio` is important only for the initialisation phase. During the calculation, the aspect ratio evolves due to non-isothermal effects and it is calculated self-consistently from the local sound speed and Keplerian frequency (see Sect. 2.1 in Chrenko et al. 2017).

- `IndirectTerm` has the same purpose to prevent undesired accelerations of a frame centered to the primary, but the term is computed in a slightly different way. This is because THORIN uses modified approach for planet-disk interactions which are treated by means of a vertically-averaged gravitational potential (Müller et al. 2012). We always set the `IndirectTerm` parameter to `YES`.

- `ExcludeHill` has the same purpose to exclude part of the gas in planet's Hill sphere from the torque computation, but the procedure follows the prescription from Crida et al. (2008) (see their Eq. 5).

- `FlaringIndex` affects only the initialisation phase. The disk should become self-consistently flared during the calculations if heating/cooling processes are accounted for.

- `Eccentricity` is a deprecated switch. Planetary eccentricities are now explicitly given for each planet in the planet configuration file (specified by the `PlanetConfig` parameter).

(Possibly) incompatible standard parameters. The following parameters are inherited from the standard FARGO code but they were not tested in THORIN. Using these parameters may lead to an unexpected behaviour and code failures:

`LabelAdvection`, `ReleaseRadius`, `ReleaseDate`, `OuterSourceMass`, `ImposeDiskDrift`, `CavityRadius`, `CavityRatio`, `CavityWidth`, `TransitionRadius`, `TransitionRatio`, `TransitionWidth`, `LambdaDoubling`.

Heating/cooling parameters. The following list summarizes and describes the model parameters associated with the energy equation:

- `EnergyEquation`; the default value is `NO`. This parameter will start the non-isothermal calculations. Please note that certain parts of the code were not adjusted for the isothermal approximation thus we advise to keep `EnergyEquation` set to `YES` all the time. Use a different version of FARGO for isothermal calculations.

- `WriteTemperature`; the default value is `NO`. The code will output temperature fields when set to `YES` and also if `EnergyEquation` is set to `YES`. The corresponding output name is `gastemper*.dat`.

- `WriteEnergy`, `WriteDivV`, `WriteQplus`, `WriteQbalance`; the default value is `NO`. By setting to `YES` you enable the output of the gas specific internal energy, velocity divergence, viscous heating term, and (viscous heating - vertical cooling) balance term, respectively. The corresponding output names are `gasenergy*.dat`, `gasdivv*.dat`, `gasqplus*.dat` and `gasqbalance*.dat`.

- `AdiabInd`; the default value is 1.4. Sets the value of the adiabatic index (the specific heat ratio) of the gas.

- `CoolingTime`; the default value is -1.0. When this parameter is set to ≤ 0.0, the energy equation will be solved implicitly (using SOR) and will automatically include the compressional heating term, the viscous heating term, the vertical cooling term and the midplane radiative diffusion. Remaining terms (stellar irradiation and accretion heating) are also included if specified by their own switches. When the parameter is set to > 0.0, it represents the characteristic cooling time which counterbalances the viscous heating and compressional heating term. No other energy source terms are accounted for in such a setup and the equation is solved using a simple numerical scheme.

- `StellarIrradiation`; the default value is `NO`. Set to `YES` in order to include the stellar irradiation heating term.

- `EffectiveTemperature`; the default value is 5656.0. Determines the effective temperature of the central protostar in Kelvins. Only affects the calculations with `StellarIrradiation` set to `YES`.

- `StellarRadius`; the default value is 3.0. Determines the radius of the protostar. Only affects the calculations with `StellarIrradiation` set to `YES`.

- `DiscAlbedo`; the default value is 0.5. Determines the albedo of the disk with respect to stellar irradiation. Only affects the calculations with `StellarIrradiation` set to `YES`.

- `OpacityDrop`; the default value is 0.6. Determines the fractional drop of the opacity in the direction perpendicular to midplane. Set to 1.0 to neglect the opacity changes. The value 0.6 was found to produce good agreement of the disk structure with 3D models.

- `ParametricOpacity`; the default value is 0.0. When the value is ≤ 0.0, the mean Rosseland opacity of the gas is calculated according to Bell & Lin (1994). When the value is > 0.0, it is used as a constant uniform opacity throughout the simulation.

Gas disk initialisation from prescribed files. With THORIN, it is possible to initialise the gas disk from prescribed ascii files. This is a useful feature which allows e.g. to start the simulation from a previously relaxed equilibrium state. In order to use this initialization:

- Set the `InitializeFromFile` parameter to `YES`;

- Specify the `DensInfile`, `VradInfile`, `VthetaInfile` and `TemperInfile` parameters. The parameters must contain valid paths to the ascii files containing the desired values of the gas surface density, radial velocity, azimuthal velocity and temperature, respectively. Each file must be a 1-column list of real numbers sorted in the same order as in a FARGO 1D hydrodynamic array (i.e. the values should be listed ring by ring, from the inner one to the outer one). Ascii files like this can be created by translating the binary outputs of hydrodynamic arrays one record after another. Such a procedure is part of the second example simulation (see the readme file).

Damping boundary condition settings. When `InnerBoundary` is set to `DAMPING`, it is possible to specify the following:

- `DampTowards`; the default value is `ZEROVRAD`. When the parameter is set to `ZEROVRAD`, the wave-killing boundary condition acts only to damp the radial gas velocity to zero. Other hydrodynamic quantities obey the closed boundary condition. The parameter can be optionally set to `INIT`. If so, the wave-killing boundary condition still damps the radial gas velocity to vanish and the remaining hydrodynamic quantities are damped towards their initial values. This is useful when the initial state corresponds to a disk relaxed by the heating/cooling processes.

- `DampingRminFrac`; the default value is 1.25. The inner wave-killing zone stretches from `Rmin` to `Rmin * DampingRminFrac`.

- `DampingRmaxFrac`; the default value is 0.84. The outer wave-killing zone stretches from (`Rmax * DampingRmaxFrac`) to `Rmax`.

- `DampingPeriodFrac`; the default vale is 1.0. The damping time scale is equal to (T_{K}`* DampingPeriodFrac`) where T_{K} is the shortest Keplerian orbital period in the respective wave-killing zone.

Parameters of the REBOUND integrator interface. THORIN automatically solves the N-body interaction among the massive bodies using the IAS15 integrator (Rein & Spiegel 2015) from the REBOUND package (Rein & Liu 2012). The following list provides parameters related to the orbital evolution of planets:

- `NoutElements`; the default value is 1. The parameter is an analogue of `Ninterm` but affects the output sampling of the file with orbital elements named `nbody.orbits.dat`. The time interval between two successive outputs of orbital elements is (`NoutElements * DT`).
- `PlanetaryDensity`; the default value is 1.0. Defines the bulk density of planets which is used to derive their radius from the planetary mass. Only important when searching for collisions among the planets.
- `ResolveCollisions`; the default value is `NO`. When set to `YES`, the REBOUND package will start to detect collisions between planets. Each collision will result in a merger.
- `TargetNpl`; the default value is -1. When set to a positive integer or zero, the code will terminate once the number of planets equals `TargetNpl`.
- `IAS15Precission`; the default value is 10^{-9}. Defines the desired precision of the IAS15 integration algorithm.
- `IAS15MinDT`; the default value is 0.0. Restricts the minimum allowed sub-step value resulting from the IAS15 adaptive time step subdivision.

Disk-planet interactions. These are several switches controlling the disk-planet interactions:

- `WriteTorqueFiles`; the default value is `YES`. In the default settings, the code writes the values of the disk torques acting on each of the planets into the files named `tqwk*.dat`. This output can be disabled by setting `WriteTorqueFiles` to `NO`.
- `HillCut`; the default value is 0.8. The parameter sets the steepness of the tapering function which excludes part of the Hill-sphere gas from the torque computation. It corresponds to the dimensionless parameter p from Eq. (5) in Crida et al. (2008). The parameter only affects the simulation if `ExcludeHill` is set to `YES`.
- `VerticalDamping`; the default value is 0.1. The parameter scales the inclination damping prescription from Tanaka & Ward (2004). It corresponds to the dimensionless parameter β from Eq. (39) in Chrenko et al. (2017). The parameter is problem-dependent and it should be tuned so that the eccentricity and inclination damping of planetary orbits operate on comparable time scales.
- `PlanetsFeelDisk`; the default value is `NO`. The parameter determines whether the planets migrate due to disk torques (when set to `YES`) or not (when set to`NO`).

- `AccretionRate`; the default value is 0.0. When set to positive value, the planets will accrete gas according to the original accretion algorithm of FARGO. We emphasize that the gas accretion algorithm was not adjusted to account for 3D orbits and it also does not cause accretion heating. We advise the users not to use it at this point.

Pebble disk parameters. In order to initialise the two-fluid calculations, in which the second fluid represents the disk of pebbles, you must first ensure that the gas disk is in a thermally relaxed equilibrium state. This is because the initial model of the pebble disk assumes a steady-state situation and it may lead to unrealistic results otherwise.

A typical procedure is to start with a preparatory simulation with the gas disk only and once the disk is relaxed, it is possible to use it as a starting point for a full simulation with the pebble disk. A simple way of doing this is with the option `InitializeFromFile` set to `YES`, provided that the `DensInfile`, `VradInfile`, `VthetaInfile` and `TemperInfile` parameters refer to the ascii files taken from the end of the preparatory simulation. The pebble disk is included into the simulation by

- setting `PebbleAccretion` to `YES` (the default setting is `NO`).

When the pebble disk is incorporated, all planets automatically start to accrete from it (but accretion heating is switch-dependent; see below).

When `PebbleAccretion` is set to `YES`, the code will automatically output several binary files, formatted as the standard FARGO polar grid arrays. There will be files written at each output, representing the pebble component surface density, the radial velocity and the azimuthal velocity. These are named `gaspebbledens*.dat`, `gaspebblevrad*.dat` and `gaspebblevtheta*.dat`, respectively. There will also be two files written only once, at the beginning of the simulation, representing the dominant pebble sizes and corresponding Stokes numbers. These are named `gaspebblesize0.dat` and `gaspebblestokes0.dat`.

The initial state of the pebble disk is described in Sect. 2.4 in Chrenko et al. (2017) and depends on the following parameters:

- `PebbleFlux`; the default value is 2×10^{-4}. The parameter defines the radial pebble mass flux in the initial steady-state and it is given in Earth masses per year ($M_{\mathrm{E}}\,\mathrm{yr}^{-1}$).

- `PebbleAlpha`; the default value is 1×10^{-4}. This is a parametrisation of the turbulent stirring of solid particles and it is usually understood as an analogue to the Shakura-Sunyaev α of the gas disk. In our model, this parameter affects the scale height of the pebble disk.

- `PebbleCoagulation`; the default value is 0.5. The parameter determines the coagulation efficiency of solids and affects the dominant size and Stokes number.

- `PebbleBulkDens`; the default value is 1.0. It represents the material density of pebbles, necessary to calculate the Stokes number in the Epstein gas drag regime.

- `SchmidtNumber`; the default value is 1.0. Only affects the simulations with `ParticleDiffusion` set to `YES` (see below). It represents the ratio of the gas diffusivity to the pebble diffusivity.

The interactions between the pebble component and the rest of the system can be tuned by the following parameters:

- `BackReaction`; the default value is `NO`. When it is set to `YES`, the drag back-reaction term is included into the Navier-Stokes equation of the gas.
- `AccretionalHeating`; the default value is `NO`. When it is set to `YES`, the energy deposited by pebbles accreting onto planets is transformed into planetary luminosity and the planets act like additional heat sources (see Chrenko et al. 2017).
- `HeatingDelay`; the default value is 100. It specifies the number of time steps `DT` over which the planetary luminosity is gradually increased from zero to the non-reduced value.
- `ParticleDiffusion`; the default value is `NO`. When it is set to `YES`, the diffusive velocity will be added to the fluid velocity of pebbles, according to Sect. 2.6 in Chrenko et al. (2017).

Finally, there is an optional output parameter related to pebbles:

- `WriteEta`; the default value is `NO`. When it is set to `YES`, the code will write the gas rotation parameter η (Eq. 22 in Chrenko et al. 2017) into `gaseta*.dat`.

Miscellaneous parameters and tools. These are the remaining parameters of THORIN:

- `ParametricAccretion`; the default value is 0.0. The parameter represents a parametric doubling time of accreting planets given in kyr. When it is set to a positive value, the planets will grow according to this doubling time. This type of accretion can heat the planets if `AccretionalHeating` is set to `YES`. This switch is useful e.g. to study the heating torque induced by a fixed accretion rate.
- `TorqueMapInfile`; the default value is `NO`. When it is set to `YES`, the code will write a file named `torquemap_infile*.dat`. The file can serve as an input file for the TORQUEMAP code written by Bertram Bitsch that is capable of producing the migration maps. The TORQUEMAP code is not included in THORIN.
- `GetTorqueForPlanet`; the default value is -1. The parameter can be set to a FARGO identification number of a planet (starting with 0). If so, the code will write a file named `gastorque*.dat`. The file contains a cell-by-cell data record of the torque acting on the planet specified by the switch. This can be helpful e.g. to calculate the radial torque density, or to plot the torque distribution over the individual cells.

2.5 Output files

The output files are written into an output directory specified by the `OutputDir` parameter. The directory is created in case it does not exist.

THORIN always outputs standard FARGO files named `dims.dat`, `used_rad.dat`. Moreover, it writes a new ascii file named `omegaframe.dat`. This file contains two columns, the first is the output number and the second is the instantaneous angular velocity of the coordinate frame.

Depending on how the `Write*` parameters are set, THORIN can output the following files, formatted as standard FARGO binary outputs of the polar grid arrays.

- Standard quantities describing the gas disk: `gasdens*.dat`, `gasvrad*.dat`, `gasvtheta*.dat`.
- New quantities describing the gas disk, already described in the previous section: `gastemper*.dat`, `gasenergy*.dat`, `gasdivv*.dat`, `gasqplus*.dat`, `gasqbalance*.dat`, `gaseta*.dat`.
- New quantities describing the pebble disk, already described in the previous section: `gaspebbledens*.dat`, `gaspebblevrad*.dat`, `gaspebblevtheta*.dat`, `gaspebblesize0.dat`, `gaspebblestokes0.dat` (the later two files are only written once at the simulation beginning).
- New quantity describing the torque from individual cells, acting on the planet specified by the `GetTorqueForPlanet` parameter: `gastorque*.dat`.

In the list above, the asterisks stand for the output numbers.

Hydrodynamic polar grids have the same output as in FARGO thus their visualisation and analysis should be straightforward for former FARGO users. For the purpose of unit conversions, please note that THORIN uses 1 AU as the length unit, 1 Solar mass as the mass unit, the gravitational constant is $G = 1$, as well as the mean molecular weight and the gas constant.

The REBOUND interface of THORIN will produce the following outputs:

- The whole REBOUND simulation is written into a binary file with the same sampling frequency as the hydrodynamic outputs (each `Ninterm * DT`). The name of the file is `nbody.simulation*.dat`. There is no need for you to analyse this file but it serves as a suitable starting point for restarts.
- Orbital elements are written into an ascii file named `nbody.orbits.dat` each (`NoutElements * DT`). Line by line, it lists the orbital elements for each planet. Each line start with the planet identification number that starts from 1 (this differs from FARGO itself which numbers the planets from 0). The ID is followed by the simulation time in code units, the semimajor axis in AU, the eccentricity and angular elements in radians (inclination, longitude of ascending node, argument of pericenter, true anomaly).

Finally, the instantaneous planet mass and radius are written and also the Cartesian coordinates (x, y, z) in AU.

- If a planet is discarded from the simulation because it was scattered (or migrated) out of the hydrodynamic domain, it is recorded in an ascii file named `nbody.discard.dat`. Each discard event has four lines. The first line contains the event time in code units; the second line contains the planet ID, mass and radius; the third line contains the final Cartesian coordinates; and the fourth line contains the final Cartesian velocities (again in the code units).
- When `ResolveCollisions` is set to `YES` and a planetary merger is detected, it is recorded in an ascii file named `nbody.mergers.dat`. Each merger event consists of $1 + 3 \times 3$ lines. The first line gives the merger time and planet IDs of both colliding planets. The first two three-line blocks describe the colliding planets, the third three-line block describes the resulting merger. Each three-line block contains the planet ID, mass and radius; the Cartesian coordinates; the Cartesian velocities.

In THORIN, the output of total disk torques acting on planets is slightly modified:

- When `WriteTorqueFiles` is set to `YES`, several ascii files will be written named `tqwk(i).dat`, where `(i)` stands for the planet ID, but now starting from 0 (this is to keep consistency with previous outputs of FARGO. Each file provides the temporal evolution of total disk torques acting on the respective planet. Each line of the output file gives the simulation time, the total specific torque acting on the planet, the normalised torque acting on the planet, midplane Cartesian coordinates (x, y) and midplane Cartesian accelerations (a_x, a_y) acting on the planet due to the disk gravity.

2.6 Restart

The restart files must be placed (or remain) inside the output directory specified by the input parametric file. In order to restart a run, you must always provide the files `dims.dat`, `omegaframe.dat` and `used_rad.dat`. To restart the non-isothermal gas disk, you must provide the files `gasdens(N).dat`, `gasvrad(N).dat`, `gasvtheta(N).dat`, `gastemper(N).dat` and `nbody.simulation(N).dat`. To restart a simulation with the pebble disk included, you must provide the files `gaspebbledens(N).dat`, `gaspebblevrad(N).dat`, `gaspebblevtheta(N).dat` and `gaspebblesize0.dat`.

The restart command is similar to the standard FARGO code:

(MPI only $\rightarrow$) `mpiexec` MPISET ($\leftarrow$ MPI only) `./thorin --s (N)` PARAM
where the `s` switch tells the code to restart from the output number given instead of `(N)`.

Bibliography

Baruteau, C. & Masset, F. 2008, ApJ, 672, 1054

Bell, K. R. & Lin, D. N. C. 1994, ApJ, 427, 987

Chrenko, O., Brož, M., & Lambrechts, M. 2017, ArXiv e-prints [[arXiv]1706.06329]

Crida, A., Sándor, Z., & Kley, W. 2008, A&A, 483, 325

de Val-Borro, M., Edgar, R. G., Artymowicz, P., et al. 2006, MNRAS, 370, 529

Klahr, H. & Kley, W. 2006, A&A, 445, 747

Lambrechts, M. & Johansen, A. 2014, A&A, 572, A107

Masset, F. 2000, A&AS, 141, 165

Müller, T. W. A., Kley, W., & Meru, F. 2012, A&A, 541, A123

Rein, H. & Liu, S.-F. 2012, A&A, 537, A128

Rein, H. & Spiegel, D. S. 2015, MNRAS, 446, 1424

Tanaka, H. & Ward, W. R. 2004, ApJ, 602, 388