GENGA

Release 2.0

Simon Grimm

Apr 17, 2026

SETUP:

Author: Simon L. Grimm
Department of Astrophysics
University of Zürich, Switzerland

GENGA is available at https://bitbucket.org/sigrimm/genga

CHAPTER

ONE

LICENSE

GENGA is free to use, but when results from GENGA are published, then the following papers have to be referenced [GrimmStadel14], [GrimmStadelBrasser+22].

CHAPTER

TWO

SETUP

2.1 GENGA Tutorial

Learn how to install and run GENGA in Google Colab with this tutorial . Open the notebook file in Google Colab and follow the tutorial step by step.

2.2 Setup

2.2.1 Requirements

Originally, GENGA was designed to run on NVIDIA GPUs. The current version supports also AMD GPUs, as well as parallel multicore CPU systems by using OpenMP.

The system requirements are:

- Nvidia GPUs:
  - CUDA toolkit
  - GPU with compute capability of 3.0 or higher
- AMD GPUs:
  - ROCm and HIP
  - python3, for translating the CUDA source code to HIP
- multicore CPUs:
  - g++ compiler
  - OpenMP
  - python3, for translating the CUDA source code to HIP

2.2.2 Install CUDA

To be able to use the code on NVIDIA GPUs, one has to install the CUDA Toolkit first as described here.

We strongly recommend using a recent CUDA version to get the full performance and correct results. If an old CUDA version is used (< CUDA 9.0) then the `def_OldShuffle` parameter in the *define.h* file must be set to 1. (See *Use old Shuffle*)

Linux

Install the gcc compiler (for example, in Ubuntu install build-essential package)
Download the CUDA toolkit from: https://developer.nvidia.com/cuda-downloads .
Install the CUDA toolkit.
Reboot
Run `nvidia-smi` to check CUDA and the available GPUs.

GCC version

It can happen that the used CUDA version needs an older GCC version than the current one on the system. In that case, either a newer CUDA version, or an older gcc version should be installed. Use the following compile option to tell CUDA to use an older GCC version (for example 7.0):

```
-ccbin=g++-7
```

2.2.3 Install ROCM and HIP for AMD GPUs

When an AMD GPU is used, then ROCM and HIP needs to be installed.

Follow the instructions on:
https://rocmdocs.amd.com/en/latest/Installation_Guide/Installation-Guide.html
to install ROCM and HIP.
Run `/opt/rocm/bin/hipconfig --full` to check the installation
Run `rocm-smi` to check the GPU
Choose the platform with either `export HIP_PLATFORM=nvidia` or `export HIP_PLATFORM=amd`

2.2.4 Determine the NVIDIA compute capability

GENGA must be compiled for a specific GPU compute capability. The compute capability corresponds to the GPU generation, a list of all NVIDIA GPUS with their compute capabilities can be found here: https://developer.nvidia.com/cuda-gpus .

The compute capability can also be checked with the provided tool `CheckGPU`:

Step 1: compile the CheckGPU code with:

```
nvcc -o CheckGPU CheckGPU.cu
```

Step 2: run:

```
./CheckGPU
```

This will list the compute capabilities of all found GPUs.

2.2.5 Compile GENGA

If an old CUDA version is used (< CUDA 9.0) then the `def_OldShuffle` parameter in the *define.h* file must be set to 1. (See *Use old Shuffle*)

The source code of GENGA and a Makefile is included in the source directory. To compile GENGA, go to the source directory and type:

```
make SM=xx
```

into a terminal, where `xx` corresponds to the compute capability of the GPU (NVIDIA) or the target ID (AMD).

Use e.g. 'make SM=60' for compute capability of 6.0, or 'make SM=65' for compute capability of 6.5.

For example use:

```
make SM=35 for Tesla K20
make SM=52 for GeForce GTX 980
make SM=60 for Tesla P100
make SM=61 for GeForce GTX 1080 ti
make SM=75 for GeForce RTX 2080 ti
make SM=86 for GeForce RTX 3090
make SM=gfx906 for AMD Radeon VII
make SM=gfx90a for AMD Instinct MI200
```

When compiling GENGA with the openGL real time visualization, go to the GengaGL directory. (See *GengaGL: Real time visualization with openGL*)

When GENGA is compiled for a newer compute capability then the GPU is able to run, then the following error message will appear by running GENGA: *FGAlloc error = 13 = invalid device symbol*.

Compile GENGA with HIP for AMD GPUs

GENGA provides a tool to translate the source code from CUDA to HIP. The HIP version can run on AMD and on NVIDIA GPUs. The translation tool is located in the HIP directory.

Run:

```
python3 GengaHIP.py
```

to translate the code. GengaHIP will copy the translated source code to the HIP directory.

Type:

```
make
```

to compile GENGA with HIP.

Compile GENGA with OpenMP for multicore CPUs

GENGA provides a tool to translate the source code from CUDA to an OpenMP CPU version. The translation tool is located in the cpu directory.

Run:

```
python3 port.py
```

to translate the code. This will copy the translated source code to the cpu directory.

Type:

```
make
```

to compile GENGA with OpenMP.

On systems with hyperthreading enabled, it can be usefull to select the desired CPU cores to run on. This can be done by typing:

```
export OMP_PLACES="{0,1,2,3, ...}"
```

to the terminal, before running the code, were the numbers indicate all the core id's that should be used.

Compile GENGA on Windows

If using Cygwin on Windows, then GENGA can be compiled the same way as in Linux with:

```
make SM=xx.
```

If using the Windows Command Prompt, type:

```
nmake -f Makefile.win SM=xx.
```

Note, that the Windows C++ compiler `cl` must be installed, and the compiler path must be loaded in the command prompt. If this is not the case, it can be loaded similar to this command:

```
call "C:\Program Files (x86)\Microsoft Visual Studio\2019\Community\Common7\Tools\
↪vsdevcmd.bat"
```

or:

```
call "C:\Program Files (x86)\Microsoft Visual Studio\2019\Community\VC\Auxiliary\Build\
↪vcvars64.bat"
```

, where the exact path and file name must eventually be changed.

2.3 Running GENGA

2.3.1 Starting GENGA

Before running GENGA, an initial conditions file must be provided, as described in *The initial conditions*. An example of an initial conditions file with 2048 planetesimals is provided in the GENGA repository (initialplanet_2048.dat).

All relevant user parameters can be set in the *param.dat* file, as described in *Parameters in the param.dat file*. An example is also provided in the source directory.

GENGA can be started with:

```
./genga [options]
```

where `[options]` are optional user parameters as described in *Console Arguments*.

When GENGA is started again in the same directory, then all files are overwritten with the new simulation data. To prevent GENGA from overwriting data, the lock.file option can be used (See *Ignore Lock File*).

CPU version

When the CPU version of GENGA is used (See *Compile GENGA with OpenMP for multicore CPUs*) then GENGA can be started with:

```
./gengaCPU [options]
```

A relevant option for multicore CPU runs is `-Nomp`, where the number of CPU cores can be set.

Note that the CPU version of GENGA does not support the Multisimulation mode. In this case every sub-simulation should just be run as an independent full simulation.

2.3.2 Restarting GENGA to continue simulations

A simulation can be restarted from each coordinate output file by using the `-R` time step console argument or by setting a restart time in the *param.dat* file. Before restarting a simulation things in the *param.dat* file can be changed if necessary, but the Output name must be the same as in the original run. To be able to restart a simulation, the corresponding coordinate output file, the corresponding line in the energy file and the time file, and if used, the corresponding aeGrid file, must exist. Note that the data in the Energy-, Collisions-, Ejections-, time- and info-files are not deleted and the new data is added at the end of these files. But the coordinate output files are OVERWRITTEN with the new data. By restarting a simulation the values of E0 and the inner Energy are read from the original run and are reused. One can also use a coordinate output file to start a new simulation run with totally different parameters by using the Output file as a new initial condition file. The `Input file Format` should then be of the form:

```
<< t i m r x y z vx vy vz Sx Sy Sz amin amax emin emax - - - - >>
```

When GENGA is restarted using the console argument "-R", it changes the entry of the Lock file named lock.dat. This file must be deleted or modified when GENGA is restarted again from the same time step. This prevents from data loss by an accidental relaunch. To ignore the lock file, the `IgnoreLockFile` flag in the *define.h* file can be set to 1.

With the restart time -1, GENGA searches automatically for the last output and restarts directly from there. To determine the last output, the last entry in the time file is used.

2.3.3 Interrupting a simulation

GENGA can be interrupted with the SIGINT signal (**Ctrl-C**, kill -2). In this case, GENGA completes the current time step and writes an additional last output. With the restart time step -1, GENGA will continue the integration starting from this output. The SIGINT signal can also be sent to GENGA when using a queuing system.

With SLURM, use `#SBATCH --signal=INT@60` in the sbatch file.

For example

```
#!/bin/bash
#SBATCH -J gengatest
#SBATCH -n 1 --gres gpu:1 -p tasna
#SBATCH --time 1-0
#SBATCH --signal=INT@60

srun genga -R -1
```

This will send the SIGINT signal 60 seconds before the end of the wall time to slurm. That allows GENGA to safe to current time step and write it into the file, before the slurm job is terminated.

Note that the `srun` command must be used for slurm to work properly.

When a slurm job must be stopped manually, the same behaviour can be achieved with:

```
scancel --signal SIGINT <jobID>
```

Other kill signals e.g SIGKILL (kill -9) or SIGTERM (kill -15) will terminate GENGA immediately, and no last output is written.

2.3.4 Using test particles

The default mode of GENGA computes the gravitational force between all pairs of particles, which leads to N^2 force calculations. When N is large, then this operation is the dominant part of the entire run time.

When small particles are used in a simulation, then it can be useful to reduce the amount of mutual force calculations between small particles. This can be done with the test particle mode option `Use Test Particles` in the *param.dat*

file, or by using the console argument `-TP 1` or `-TP 2`.

Test particles are bodies which have a smaller or equal mass than the value specified in the `Particle Minimum Mass` parameter.

GENGA supports two different test particles modes, which are described below and visualized in Fig. 2.1

Test particles mode 1

`Use Test Particles = 1` Small particles (with a mass smaller than `Particle Minimum Mass`, do not interact with other particles (small and large). Large particles interact with all other large particles, and affect small particles. Small particles can collide with large bodies, but they do not perturb them gravitationally.

Test particles mode 2 (semi active)

Small particles (with a mass smaller than `Particle Minimum Mass`, do not interact with other small particles. Small particles interact with all large particles. Large particles interact with all large and small particles.

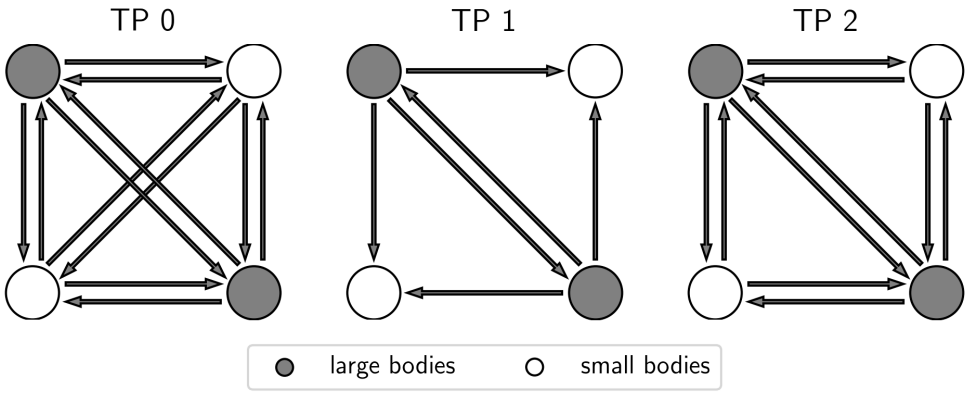

Fig. 2.1: Calculated force terms of the different test particles modes, (0, 1 and 2) for an example of two large and two small particles

2.3.5 Multi simulation mode

The multi simulation mode can be used to simulate a large number of small simulations with up to 32 massive particles. Individual sub simulations can have a different number of bodies. For each simulation a new directory is needed, containing the initial condition file and the *param.dat* file. Note that not all parameters in the *param.dat* file can be chosen individually, these are only:

- the time step
- the number of integration steps
- the name
- the central mass
- the star radius
- the star Love number
- the star fluid Love number
- the star tau

- the star spin_x
- the star spin_y
- the star spin_z
- the star Ic
- the n1 parameter
- the n2 parameter
- the initial condition file
- the default rho
- the Minimum number of bodies number
- the Minimum number of test particles number
- the inner truncation radius
- the outer truncation radius

All the other parameters are read from the simulation number 0. To start a multi simulation run, an additional file is needed, which contains a list of all sub simulation directory names.

For example a file named `path.dat`:

```
sim0000
sim0001
sim0002
.
.
.
```

The simulation can then be started with:

```
./genga [options] -M path.dat
```

where `[options]` are optional user parameters as described in *Console Arguments*.

If a sub simulation contains less particles than specified in the `Minimum number of bodies` or the `Minimum number of test particles` number, then this specific simulation is stopped and the total number of simulations is reduced.

In the multi simulation mode, the indexes of the particles should not be greater than 100.

2.3.6 Generate Keplerian-elements output files

Output files can be converted into Keplerian-elements output files with the `KE` tool, located in the `tools` directory.

Compile with typing:

```
make
```

into a terminal.

Then run `KE` with:

```
./KE [options]
```

where `[options]` are the following optional user parameters:

- `-tmin`: starting time step, (used for FormatT = 0, FormatP = 1).

- `-tmax`: ending time step ,(used for FormatT = 0, FormatP = 1).
- `-step`: output interval, (used for FormatT = 0, FormatP = 1).
- `-pmin`: starting particle index, (used for FormatT = 1, FormatP = 0).
- `-pmax`: ending particle index, (used for FormatT = 1, FormatP = 0).
- `-in`: name of the files
- `-Msun`: Solar Mass (default = 1.0 $M_\odot$)

Alternative to the console options, a parameter file called `paramKE.dat` can be used. This file contains the following arguments:

- `Output name`: name of the files
- `-tmin`: starting time step, (used for FormatT = 0, FormatP = 1).
- `-tmax`: ending time step ,(used for FormatT = 0, FormatP = 1).
- `-step`: output interval, (used for FormatT = 0, FormatP = 1).
- `-pmin`: starting particle index, (used for FormatT = 1, FormatP = 0).
- `-pmax`: ending particle index, (used for FormatT = 1, FormatP = 0).
- `-Central Mass`: Solar Mass (default = 1.0 $M_\odot$)
- `Output file Format:` This must correspond to the entry of the GENGA `param.dat` file.

The KE tool generates for each coordinate output file a corresponding Kepler-Elements file, see *Keplerian-Elemets Output Files: aei<name>.dat*.

Example for FormatT = 0, FormatP = 1 (default):

```
./KE -tmin 0 -tmax 1000 -step 10 -in test
```

This example will generate Kepler-Element files from timestep 0 to 1000 in output intervals of 10 for a simulation name 'test'.

Example for FormatT = 1, FormatP = 0:

```
./KE -pmin 0 -pmax 2048 -in test
```

This example will generate Kepler-Element files from particle index 0 to 2048 for a simulation name 'test'.

2.3.7 Convert output files to barycentric coordinates

Heliocentric output files can be converted into barycentric output files with the `ConvertHelioToBarry` tool, located in the `tools` directory.

This tool works only in the output format, FormatT = 0, FormatP = 1. If a different output format is uses, then the conversion tools should be used to convert before and after this step (*Convert_P1T0_to_P0T1* and *Convert_P0T1_to_P1T0*).

Compile with typing:

```
make
```

into a terminal.

Then run `ConvertHelioToBarry` with:

```
./ConvertHelioToBarry [options]
```

where [options] are the following optional user parameters:

- -tmin: starting time step, (used for FormatT = 0, FormatP = 1).
- -tmax: ending time step ,(used for FormatT = 0, FormatP = 1).
- -step: output interval, (used for FormatT = 0, FormatP = 1).
- -in: name of the files
- -Msun: Solar Mass (default = 1.0 $M_{\odot}$)

The ConvertHelioToBarry tool generates for each coordinate output file a corresponding barycentric output file, see *Barycentric output Files: OutBary<name>.dat*.

Example:

```
./ConvertHelioToBarry -tmin 0 -tmax 1000 -step 10 -in test
```

This example will generate barycentric output files from timestep 0 to 1000 in output intervals of 10 for a simulation name 'test'.

2.3.8 Convert_P0T1_to_P1T0

This tool converts GENGA outputs from FormatP = 0 & FormatT = 1, to FormatP = 1 & FormatT = 0.

Compile with typing:

```
make
```

into a terminal.

Then run Convert_P0T1_to_P1T0 with:

```
./Convert_P0T1_to_P1T0 [options]
```

where [options] are the following optional user parameters:

- -pmin: starting particle index.
- -pmax: ending particle index.
- -in: name of the files
- -dt: time step in days (default = 6.0 days)
- -t: Start time of the simulation in years (default = 0)

Example:

```
./Convert_P0T1_to_P1T0 -pim 0 -pmax 1000 -in test -dt 6 -t 0
```

This example will generate output files containing all particles with indices between 0 and 1000, for a simulation name 'test' and a used time step of 6 days and a starting time of the simulation of 0 years.

2.3.9 Convert_P1T0_to_P0T1

This tool converts GENGA outputs from FormatP = 1 & FormatT = 0, to FormatP = 0 & FormatT = 1.

Compile with typing:

```
make
```

into a terminal.

Then run `Convert_P1T0_to_P0T1` with:

```
./Convert_P1T0_to_P0T1 [options]
```

where `[options]` are the following optional user parameters:

- `-tmin`: starting time step.
- `-tmax`: ending time step.
- `-step`: output interval.
- `-in`: name of the files

Example:

```
./Convert_P1T0_to_P0T1 -tim 0 -tmax 1000 -step 10 -in test
```

This example will generate output files containing all particles between the time steps 0 and 1000 with a time step output interval of 10 and a simulation name 'test'.

2.3.10 Cleaning the time files

The time file contains the time in seconds, GENGA needed to reach the next coordinate interval. When the code was interrupted in between of two output intervals, then the time file contains additional lines with the divided times in it. If the code was interrupted many times due to e.g. limited wall times, the time files can get hard to read.

The tool `cleanTime.py` in the `tools` directory can be used to clean the time files and add the divided times together to the original coordinate output intervals.

The tool can be used with:

```
python3 cleanTime.py -n <name> -ci <I>
```

where <name> is the name in the simulation files, and <I> is the coordinate output interval used in the param.dat file.

The tool will create a new file `timeClean<name>.dat` with the cleaned times.

2.3.11 Cleaning the Energy files

Similarly to the time files, the Energy files contain additional lines when a run was interrupted. With the tool `cleanEnergy.py` in the `tools` directory, the Energy files can be cleaned.

The tool can be used with:

```
python3 cleanEnergy.py -n <name> -ei <I> -dt <timestep>
```

where <name> is the name in the simulation files, <I> is the energy output interval, and <timestep> is the time step used in the param.dat file.

The tool will create a new file `EnergyClean<name>.dat` with the cleaned Energies.

2.3.12 Irregular output times

When output data is needed on an irregular interval, then the `Coordinates output interval` and `Energy output interval` parameters are not useful. Instead a calendar file with the desired output times can be provided. The name of this file must be set in the `Irregular output calendar` parameter in the *param.dat* file.

The file must contain line by line the times of the desired outputs in units of years. At each irregular output time, and entry in the irregular energy file is written, and a new irregular output file is created (*The Irregular Coordinate Output Files: OutIrr<name><time>.dat* and *The Irregular Energy Output File: EnergyIrr<name>.dat*).

For example, a calendar file containing the following lines:

```
0.1
0.11
0.3
```

creates three irregular output files OutIrrtest_000000000000.dat, OutIrrtest_000000000001.dat and OutIrrtest_000000000002.dat.

When starting a new simulation, then old OutIrr<name>.dat files and EnergyIrr<name>.dat files are not deleted. If the EnergyIrr<name>.dat file already exists, then the initial energy for a new simulation is read from this file.

When the multi simulation mode is used, then the time and time-step information is only read from the first sub-simulation and applied to all simulations synchronously.

The number of digits in the output filenames can be changed with the `def_NFileNameDigits` parameter in the *define.h* file

2.3.13 Use self tuning kernel parameters

GPU kernels need to be configured with kernel parameters. These are the number of threads per threadblock and the number of threadblocks. The performance of a GPU code can depend on the specified parameters. Also depending on the used initial conditions and the used GPU type, the best choice of the kernel parameters can be different. Therefore GENGA uses a self tuning routine to determine the best choice at the beginning of the simulations. The used parameters are reported in *The tuningParameters.dat file*. If the tuning routine is not enabled, then this file can be used to set the parameters. If the file does not exist and if the tuning routine is enabled, then default values are used for the parameters.

If *Exact reproducible results* is used, then always the default values are used. The reason is that a different choice of kernel parameters can lead to a different rounding error array summations.

When the performance of GENGA measured with a profiling tool (e.g nvprof or nsys) then, GENGA should be run first with the tuning routine enabled, to write the *The tuningParameters.dat file*. And in a second step, the profiling can be done without the tuning routine. The reason for this is that the tuning routine is running some kernels many times, which affects the profiling statistics.

The tuning routine can be enabled with the `Do kernel tuning` parameter in the *param.dat* file.

2.3.14 GengaGL: Real time visualization with openGL

GENGA offers the option to view a real time visualization of a simulation by using openGL. Thi option can only be used with a GPU that is connected directly to the monitor. Typically, GPUs in a computer cluster can not be used for this. GENGA uses the CUDA openGL interoperability to visualize the simulated particle directly with the GPU. No data transfer to the CPU is needed, therefore only very little extra run time is added to the simulation.

To compile GENGA with openGL, GLUT must be installed. In Ubuntu this can be installed with the freeglut3-dev package:

```
sudo apt install freeglut3-dev
```

The GENGA - openGL interoperability code is included in the GengaGL directory and can be compiled with the provided Makefile in the same way as the original GENGA code.

GengaGL can be started with:

```
./gengaGL [options]
```

where [options] are optional user parameters as described in *Console Arguments*.

When GengaGL is started, then a window with the visualization is created. At the origin, the axis of the coordinate system is plotted. On the bottom left, the time of the integration is shown. On the bottom right, the scale of the coordinate system axis is indicated. The color of the particles correspond to the masses of the bodies. Red indicates the largest masses, yellow the smallest masses. Massless particles are plotted in white color.

The following options can be used to interact with the visualization:

- Left click and move the mouse to rotate around the z-axis and the y-axis.
- Right click and move the mouse up or down to change the rotation speed of the reference frame.
- Press 'a' + left click and move the mouse to shift the origin.
- Press space, 'p' or middle click to pause the simulation. Press again to continue.
- Use the scroll option to zoom in and out of the visualization.
- Press 'r' to increase the point size of the particles.
- press 't' to decrease the point size of the particles.
- Press 'o' to reset to the original visualization settings.

In Fig. 2.2 is shown a screenshot of GengaGL.

2.4 The initial conditions

The following parameters are relevant for the initial conditions and can be set in the *param.dat* file:

- `Input file`
- `Input file Format`
- `Default rho`
- `Angle units`

The initial conditions must be provided in a file, and the name of this file must be set with the `Input file` argument. The file must be a text file and every particle corresponds to a new line in the file. The central mass (Sun) must not be included in the initial conditions. Values for the central mass can be specified directly in the *param.dat* file, the position and the velocities of the central mass are set to the origin (heliocentric coordinates).

The initial conditions must be a text file and the format must correspond to the values set in the *param.dat* file (Input file Format: `<< ... >>`). The data of each particle has to be written in a new line in text format, and the format of the data must correspond to the values of `Input file Format`.

2.4.1 The initial conditions format

The format of the initial conditions file must be specified with the `Input file Format` option. The different entries must be given between the << and the >> characters and a blank space must be included between every entry. All available options are listed below. Most values are optional, but they must contain a complete set of Cartesian or Keplerian elements, e.g a full set of [x y z vx vy vz] or [a(or P) e inc O w M(orT)]

Possible arguments are:

- t: start time of the simulation, in years (default = 0.0).

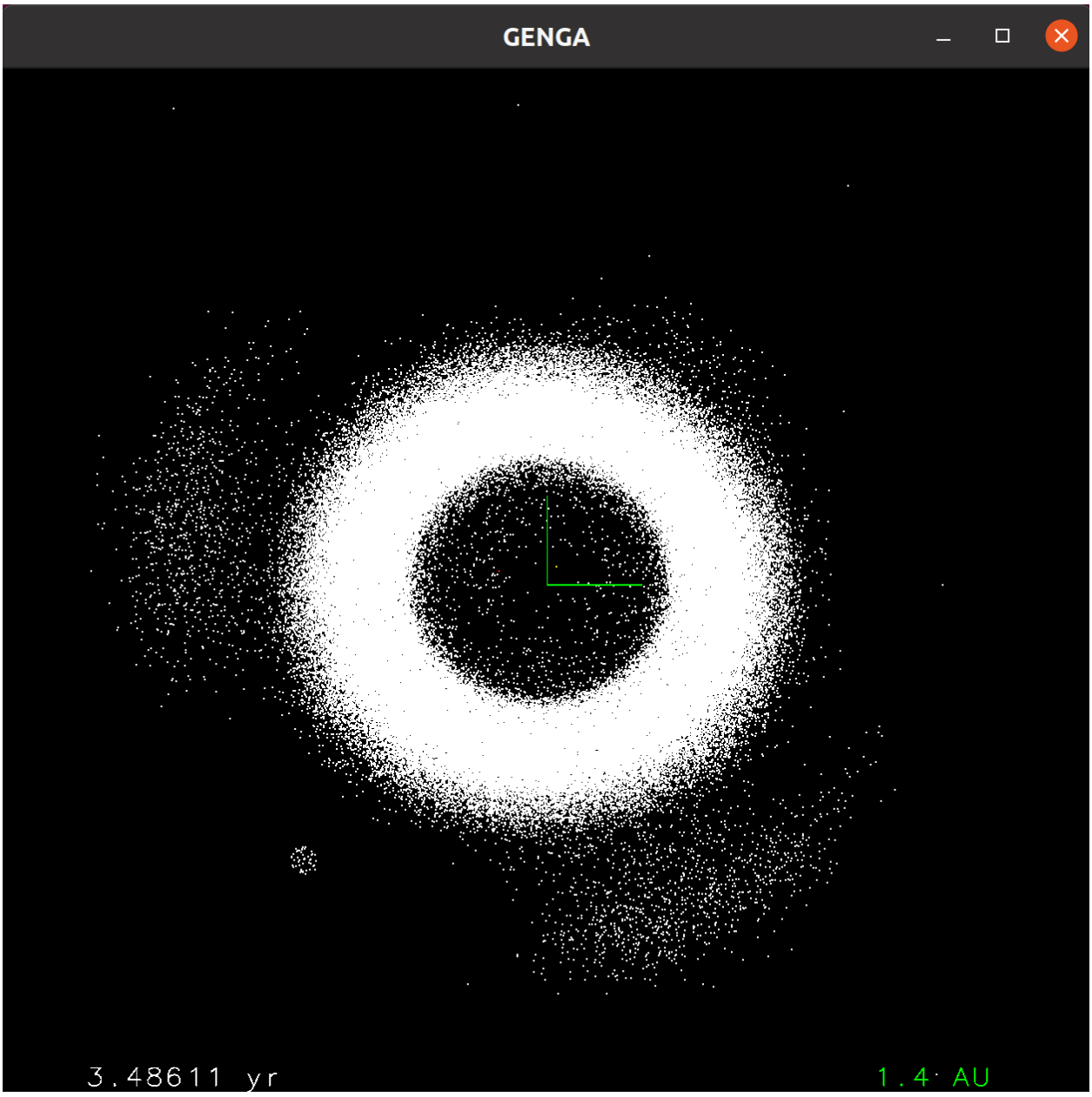

Fig. 2.2: Screenshot of the GENGA real time visualization tool using openGL. Shown is the inner part of the Solar System, including Jupiter and the asteroid belt. The Trojans are nicely visible on the Lagrange points of Jupiter.

- i: index of the body. The default value is the line number in the input file, starting with 0. Indices should be unique.
- m: mass in Solar masses (default = 0.0).
- r: physical radius in AU. If r is not given or the radius is equal to zero, then the program uses the density to calculate the radius.
  Note that there are different ways to set a radius or density. (See *Examples*).
- rho: density in g/cm^3; optional. The default value can be set by the `Default rho` parameter.
- x: x-position in AU (heliocentric).
- y: y-position in AU (heliocentric).
- z: z-position in AU (heliocentric).
- vx: x-velocity in AU/day * 0.0172020989 (heliocentric) (See *Units*).
- vy: y-velocity in AU/day * 0.0172020989 (heliocentric) (See *Units*).
- vz: z-velocity in AU/day * 0.0172020989 (heliocentric) (See *Units*).
- a: semi-major axis in AU.
- P: period in days.
- e: eccentricity.
- inc: inclination in radians or degrees (See `Angle units`).
- O: (Omega, Ω) longitude of the ascending node, in radians or degrees (See `Angle units`).
- w: (omega, ω) argument of periapsis, in radians or degrees (See `Angle units`).
- M: mean anomaly, in radians or degrees (See `Angle units`).
- T: Time of first transit, in BJD (See *Units*).
- Sx: x-spin in Solar masses AU^2 / day * 0.0172020989. (default = 0.0) (See *Units*).
- Sy: y-spin in Solar masses AU^2 / day * 0.0172020989. (default = 0.0) (See *Units*).
- Sz: z-spin in Solar masses AU^2 / day * 0.0172020989. (default = 0.0) (See *Units*).
- amin: minimal value of semi major axis range for aecount; optional (default = 0.0). (See *aeLimits*).
- amax: maximal value of semi major axis range for aecount; optional (default = 100). (See *aeLimits*).
- emin: minimal value of eccentricity range for aecount; optional, (default = 0.0). (See *aeLimits*).
- emax: maximal value of eccentricity range for aecount; optional, (default = 1.0). (See *aeLimits*).
- k2: potential Love number of degree 2, dimensionless (default = 0.0).
  If this is given, then it must also be set in the `Output file Format`: argument.
- k2f: fluid Love number of degree 2, dimensionless (default = 0.0).
  If this is given, then it must also be set in the `Output file Format`: argument.
- tau: time lag in day / 0.0172020989 (See *Units*) (default = 0.0).
  If this is given, then it must also be set in the `Output file Format`: argument.
- Ic: moment of inertia, dimensionless (See *Units*) (default = 0.4).
  If this is given, then it must also be set in the `Output file Format`: argument.
- Rc: Critical radius for close encounters, rcrit, in AU (default = 0.0)
- test: Optional value to store in the test arrays.

- -: skip column, optional.

2.4.2 Examples

Example 1:

```
format in 'param.dat': << x y z m vx vy vz r >>
input file:
    x1 y1 z1 m1 vx1 vy1 vz1 r1
    x2 y2 z2 m2 vx2 vy2 vz2 r2
    .
    .
    .
    xn yn zn mn vxn vyn vzn rn
```

GENGA reads the radii from the initial condition file.

Example 2:

```
format in 'param.dat': << x y z m vx vy vz rho >>
input file:
    x1 y1 z1 m1 vx1 vy1 vz1 rho1
    x2 y2 z2 m2 vx2 vy2 vz2 rho2
    .
    .
    .
    xn yn zn mn vxn vyn vzn rhon
```

GENGA reads the densities from the initial condition file and computes the radii.

Example 3:

```
format in 'param.dat': << x y z m vx vy vz >>
Default rho = 2.0
input file:
    x1 y1 z1 m1 vx1 vy1 vz1
    x2 y2 z2 m2 vx2 vy2 vz2
    .
    .
    .
    xn yn zn mn vxn vyn vzn
```

GENGA uses the default density from the *param.dat* file and computes the radii.

Example 4:

```
format in 'param.dat': << x y z m vx vy vz r rho >>
Default rho = 2.0
input file:
    x1 y1 z1 m1 vx1 vy1 vz1 r1 rho1
    x2 y2 z2 m2 vx2 vy2 vz2 r2 rho2
    .
    .
    .
    xn yn zn mn vxn vyn vzn r2 rho2
```

GENGA reads the radii from the initial condition file. If a radius is set to zero, then GENGA reads the density from the initial condition file and computes the radius.

2.5 Units

The units in GENGA are chose such that the solar mass $M_\odot = 1$, the gravitational constant $G = 1$ and the distance from the Sun to Earth $AU = 1$.

With $G = 6.67408 \cdot 10^{-}11\text{m}^3\text{kg}^{-1}\text{s}^{-2} = 0.01720209895^2\text{AU}^3\text{M}_\odot^{-1}\text{day}^{-2}$, and the Gaussian gravitational constant $k = 0.01720209895$.

Therefore, we need

$$G = 0.01720209895^2\text{AU}^3\text{M}_\odot^{-1}\text{day}^{-2} = 1.0\,\text{day'}^{-2}$$

and

$$1\,\text{day'} = \text{day}/0.01720209895,$$

That means that all time units must be rescaled with 0.01720209895.

2.5.1 Time

The time units are given now as $[time] = \text{day}/0.01720209895 = \text{day'}$.
To convert time from day to day', it must be multiplicated by 0.01720209895.

Example: 1 min = 0.0006944 day = 0.00001194 day'.

An exception is the time step, it is given in days, and converted internally to code units.

2.5.2 Masses

Masses are given in Solar masses ($M_\odot$).

Example: The Sun has a mass of 1 $M_\odot$.
The Earth has a mass of $3.003 \cdot 10^{-6} M_\odot$.

2.5.3 Distances, Radii

All distances and radii are given in Astronomical Units, AU.

Example: The distance from the Sun to the Earth is 1 AU.
The Earth radius is $4.2635 \cdot 10^{-5}$ AU.

2.5.4 Velocities

[v] = AU / day' = AU / day * 0.01720209895.
To convert velocities from AU / day to AU / day', they must be divided by 0.01720209895.

Example: The Earth's orbital velocity is 30 km / s = 0.01720209895 AU / day.
In GENGA units, this is 1.0 AU / day'.

2.5.5 Spin

[spin] = $M_\odot AU^2/day' = M_\odot AU^2/day \cdot 0.01720209895$
To convert the spin from $M_\odot AU^2/day$ to $M_\odot AU^2/day'$, it must be divided by 0.01720209895.
The spin is computed as

$$\vec{S} = \vec{\Omega} \cdot I,$$

with the angular rotation rate $\vec{\Omega} = \frac{2\pi}{P}$ the rotation period P (in units of day') and the moment of inertia I:

$$I = I_c m r^2$$

The parameter I_c defines the inner structure of the body. $I_c = 2/5$ is a solid sphere with uniform density.

Example: The Sun has a rotational period P of 27.5 days, a mass of 1.0 $M_\odot$ and a radius of 0.00464 AU. We assume $I_c = 0.07$.

$$S = \frac{2\pi}{27.5\text{days} \cdot 0.01720209895} \cdot 0.07 \cdot 1.0 M_\odot \cdot (0.00464 AU)^2 = 0.00002001703\, M_\odot AU^2/day'$$

2.5.6 Energy

The energy in the Energy file is reported in in $M_\odot AU^2/day^2$.

2.5.7 Angular momentum

The angular momentum in the Energy file is reported in in $M_\odot AU^2/day$.

2.5.8 Time lag

The time lag τ is given in units of day' = day / 0.0172020989.
To convert τ from day to day', it must be multiplicated by 0.01720209895.

Example: 1 min = 0.0006944 day = 0.00001194 day'.

The time lag τ is related (approximately) to the tidal quality factor Q via [Efroimsky07]:

$$\tau = \frac{\arctan(1/Q)}{2|n - \omega|},$$

with the mean motion of the planet n and the rotation rate of the star ω.

2.5.9 Time of first transit

The time of first transit is given in Barycentric Julian Date (BJD).

2.5.10 Physical constants

The values of used physical constants are set in the *define.h* file.

2.6 Parameters in the param.dat file

All parameters in the *param.dat* file can be changed with recompiling GENGA. Most parameters are optional, if they are not included in the *param.dat* file, then their default values are used. The default values are set in the *define.h* file.

2.6.1 Main parameters

- `Time step in days`, (default = 6.0 days)
  Should be minimal ca. 30 steps per orbit for the second order integrator.
  Can be set to negative values for an integration backward in time.
- `Output name` (default = "test").
  All output files will contain this name,
- `Energy output interval`, Interval when energy information is written, in units of timesteps, (default = 100).
  See *The Energy Output File: Energy<name>.dat*
  - when set to 0, then no energy output is written.
  - when set to -1, then only the very last energy output is written (or when a simulation is stopped).
- `Coordinates output interval`, Interval when output coordinates are written to a file, in units of timesteps, (default = 100).
  See *The Coordinate Output Files: Out<name>.dat*
  - when set to 0, then no coordinate output is written.
  - when set to -1, then only the very last coordinate output is written (or when a simulation is stopped).
- `Number of outputs per interval`, Can be used to write outputs at a number of consecutive time steps at each output Interval.
  In units of time steps.
  See *Number of outputs per interval*.
  (default = 1).
- `Coordinate output buffer`, can be used to temporarily store outputs in a GPU buffer before transferring the data back
  to the CPU. When outputs are written frequently, then this option can speed up the data transfer.
  In units of time steps.
  (default = 1)
  The energy outputs within a buffer size are skipped in this mode.
- `Irregular output calendar`, filename of the irregular output calendar file,
  Can be use to write coordinate and energy outputs at specified times.
  See *Irregular output times*
  - - : nothing happens
  - <calendar file name>: the calendar file is used and irregular output files are written.
- `Integration steps`, total number of time steps to run.
  (default = 1000 time steps)
- `Input file`, (default = 'inital.dat')
  Filename (or path) of the initial conditions file
  The file must exist.
  See *The initial conditions*
- `Input file Format`: Format of the initial conditions file.
  (default = << x y z m vx vy vz >>)
  See *The initial conditions format*
- `Output file Format`: Format of the coordinate output files.
  (default = << t i m r x y z vx vy vz Sx Sy Sz amin amax emin emax aec aecT encc test >>)

See *Output File Format*

- `Angle units`, unit of angles in the initial conditions file, (default = radians)
  either 'radians' or 'degrees'.
  affects inc, O, w and M.
- `Use output binary files`, text or binary file format for coordinate output files.
  - 0 (default): Use text files for coordinate output files.
  - 1: Use binary files for coordinate output files.
  See *The Coordinate Output Files: Out<name>.dat*
- `Default rho`, value for the densities when no value (no rho and no r) is given in `Input file Format`:
  in g/cm^3 (default = 2.0)
  See *The initial conditions format*
- `Restart timestep`, can be used to continue a finished simulation or the restart at a given time step
  - 0: Start a new simulation, old files are overwritten (default).
  - > 0: Restart GENGA at this time step.
  - -1: Continue at the last written output time step.
  See *Restarting GENGA to continue simulations*

2.6.2 Stellar parameters

- `Central Mass`
  Mass of the central star, in Solar Masses (default = 1.0)
- `Star Radius`
  Physical radius of the central star in AU (default = 0.00465475877 AU = Solar radius)
- `Star Love Number`
  Love number of the central star (default = 1.0)
- `Star fluid Love Number`
  Fluid Love number of the central star (default = 1.0)
- `Star tau`
  Time lag for tidal force of the central star (default = 0.0)
- `Star spin_x`
  X- component of the spin of the central star, in $M_{\odot}AU^2/day \cdot 0.01720209895$ (default = 0.0)
  See *Spin*
- `Star spin_y`
  Y- component of the spin of the central star, in $M_{\odot}AU^2/day \cdot 0.01720209895$ (default = 0.0)
  See *Spin*
- `Star spin_z`
  Z- component of the spin of the central star, in $M_{\odot}AU^2/day \cdot 0.01720209895$ (default = 0.0)
  See *Spin*
- `Star Ic`
  Moment of inertia of the central star, dimensionless (I/(MR^2) (default = 0.4)
  This is used to convert between spin and rotational period.
  See *Spin*
- `J2`
  J2 value for additional gravitational multipole expansion (default = 0.0)

See *J2: additional gravitation multipole expansion*

- `J2 radius`
  Mean radius of mass distribution for additional gravitational multipole expansion, in AU (default = 0.0)
  See *J2: additional gravitation multipole expansion*

2.6.3 Integrator options

- `n1`: Parameter to set critical radius for close encounters (default = 3.0).
  See *The n1 and n2 values*
- `n2`: Parameter to set critical radius for close encounters (default = 0.4).
  See *The n1 and n2 values*
- `Use Test Particles`, flag to enable test particle mode (default = 0)
  See *Using test particles*, and `Particle Minimum Mass`.
  - 0: full gravity mode, compute force between all pairs of particles.
  - 1: test particle mode, small bodies do not affect other bodies.
  - 2: semi active mode, small bodies do only affect large bodies but not other small bodies.
- `Particle Minimum Mass`, threshold mass between small and large particles (default = 0.0)
  All particles with a smaller mass than this value are treated as test particles (if `Use Test Particles > 0`).
  When `Use Test Particles = 0`, then this parameter has no affect.
- `Symplectic recursion levels`, number of symplectic levels in the hybrid symplectic integration method (default = self tuned).
  - -1: Use self tuning routine at integration start, to find the fastest option between levels, 1, 2 or 3.
  - > 0: Use this number of symplectic recursion levels.

  See *Higher level changeover functions*.

- `Symplectic recursion sub steps`, number of sub steps per symplectic level in the hybrid symplectic integration method (default = self tuned).
  - -1: Use self tuning routine at integration start, to find the fastest option between sub steps, 2, 4, 8 or 10.
  - > 1: Use this number of sub steps per symplectic level.

  See *Higher level changeover functions*.

- `Minimum number of bodies`, (default = 0)
  When the number of bodies (not including test particles) gets smaller than this number, then the simulation will be stopped.
- `Minimum number of test particles` (default = 0)
  when the number of test particles gets smaller than this number, then the simulation will be stopped.
- `Inner truncation radius`, in AU (default = 0.2)
  When the distance of a particle to the central mass is smaller than this number, then the particle is removed from the simulation.

See *Ejections*

- `Outer truncation radius` in AU (default = 50.0)
  When the distance of a particle to the central mass is larger than this number, then the particle is removed from the simulation.
  See *Ejections*
- `Order of integrator`, set the order of the symplectic integrator.
  See *The order of the symplectic integrator*
  - 2: second order (default)
  - 4: fourth order
  - 6: sixth order

2.6.4 Memory options

- `Maximum encounter pairs` arrays size to store close encounter pairs for each body (default = 512).
  See *Close Encounters Pairs*
- `Nfragments`: Number of additional memory size for debris particles, in particle numbers, (default = 0).
  See *Model for small bodies collisions*

2.6.5 Options for the a-e and a-i grid

See *The a-e and a-i grid*.

- `Use aeGrid`, flag to enable the a-e and a-i grids, (default = 0)
  - 0: nothing happens
  - 1; a-e and a-i grids are created
- `aeGrid amin`, minimal value of the same-major axis dimension, in AU.
  (default = 0.0)
- `aeGrid amax`, maximal value of the same-major axis dimension, in AU.
  (default = 5.0)
- `aeGrid emin`, minimal value of the eccentricity dimension.
  (default = 0.0)
- `aeGrid emax`, maximal value of the eccentricity dimension.
  (default = 1.0)
- `aeGrid imin`, minimal value of the inclination dimension, in radians.
  (default = 0.0)
- `aeGrid imax`, maximal value of the inclination dimension, in radians.
  (default = 0.1)
- `aeGrid Na`, number of points in the same-major axis dimension.
  (default = 10)
- `aeGrid Ne`, number of points in the eccentricity dimension.
  (default = 10)
- `aeGrid Ni`, number of points in the inclination dimension.
  (default = 10)
- `aeGrid Start Count`, starting time step when a-e and a-i grid start, in units of time steps

(grids will start at the next bigger coordinate output step)
(default = 0)

- `aeGrid name`, name of the grid files.
  (default = A)

2.6.6 Options for the gas disk

See *Gas disk*

- `Use gas disk`: Flag to enable the gas disk. Individual effects can be selected with the following parameters.
  - 0 (default): Do not use a gas disk.
  - 1: Use a gas disk with the following parameters.
- `Use gas disk potential`: Flag to enable the gas disk potential effect.
  - 0: Do not use the gas disk potential effect
  - 1 (default): apply the gas disk potential to all particles
  - 2: apply the gas disk potential only to particles with m < `Gas Mgiant`
- `Use gas disk enhancement`: Flag to enable the gas disk enhancement effect.
  - 0 (default): Do not use the gas disk enhancement.
  - 1: apply the gas disk enhancement to all particles.
  - 2: apply the gas disk enhancement only to particles with m < `Gas Mgiant`
- `Use gas disk drag`: Flag to enable the gas drag.
  - 0: Do not apply the gas drag.
  - 1: apply the gas drag to all particles.
  - 2 (default): apply the gas drag only to particles with m < `Gas Mgiant`
- `Use gas disk tidal dampening`: Flag to enable gas disk tidal dampening (Type I migration).
  - 0: Do not use tidal dampening.
  - 1: apply tidal dampening to all particles.
  - 2 (default): apply tidal dampening only to particles with m < `Gas Mgiant`.
- `Gas disk inner edge`: The inner edge of the gas disk in AU.
  (default = 0.1 AU)
- `Gas disk outer edge`: The outer edge of the gas disk in AU.
  (default = 35.0 AU)
- `Gas disk grid outer edge`: The outer edge of the gas disk grid in AU.
  (default = 15.0 AU)
- `Gas disk grid dr`: The r-spacing of the gas disk grid in AU.
  (default = 0.1 AU)
- `Gas dTau_diss`: The dissipation time for the gas disk in years.
  (default = 10000 yr)
- `Gas Sigma_10`: The gas surface density at 1 AU, in g/cm^3.
  (default = 2000 g/cm^3)

- `Gas alpha`: The power law exponent for the gas disk surface density.
  (default = 1).
- `Gas beta`: The power law exponent for the gas disk scale height.
  (default = 0.25).
- `Gas Mgiant`: Mass limit for gas effects, in Solar masses. If m > Mgiant, the gas drag, gas potential and tidal dampening is not applied to that particle.
  (default = 1.0E-4).
- `Gas file name`: Optional filename for using individual gas disk structures.
  - '-' (No file name specified, default). Use the gas disk with the above specified parameters.
  - else: The specified file is read to set the gas disk parameters (time, r, Sigma and h).

2.6.7 Non-Newtonian forces

- `Solar Constant` : Solar constant at 1 AU in W / m^2 (default 1367.0)
- `Use GR`: Flag to enable General Relativity corrections
  See *General Relativity Corrections*
  - 0 (default): no GR correction
  - 1: use GR Hamiltonian splitting
  - 2: use implicit midpoint with GR force
  - 3: use GR force directly (not symplectic)
- `Use Tides`: Flag to enable tidal forces
  See *Tidal Forces*
  - 0 (default): no tidal forces
  - 1: use tidal forces
- `Use Rotational Deformation`: Flag to enable rotational deformation forces
  See *Rotational Deformation*
  - 0 (default): no rotational deformation force
  - 1: use rotational deformation force
- `Use force` : Old parameter to enable GR, tidal or rotational deformation.
  This parameter is outdated, use `Use GR`, `Use Tides` or `Use Rotational Deformation` instead.
  - 0 (default) no force applied
  - 1: Use GR correction with Hamiltonian splitting
  - 2: Use tidal forces
  - 4: Use rotational deformation
  - 3: GR + tidal force
  - 5: GR + rotational deformation
  - 6: Tidal force + rotation deformation
  - 7: GR + tidal force + rotation deformation
- `Use Yarkovsky`: Flag for Yarkovsky effect
  See *Yarkovsky effect*

  - 0 (default): no Yarkovsky effect
  - 1: use Yarkovsky effect $\mathbf{a_Y}$
  - 2: use time averaged Yarkovsky effect $\frac{da}{dt}$

- `Yarkovsky Interval`, in time steps: Interval in which the Yarkovsky model is called (default 1).
  See *Yarkovsky effect*
- `Use Poynting-Robertson`: Flag for Poynting-Robertson effect
  See *Poynting-Robertson effect*
  - 0 (default) : no Poynting-Robertson effect
  - 1: use Poynting-Robertson effect $\mathbf{a_{PR}}$
  - 2: use time averaged Poynting-Robertson effect $\frac{da}{dt}$ and $\frac{de}{dt}$
- `Poynting-Robertson Interval`, in time steps: Interval in which the Poynting-Robertson model is called (default 1).
  See *Poynting-Robertson effect*
- `Radiation Pressure Coefficient Qpr`, used in the Poynting-Robertson effect, in general assumed to be 1.
  See *Poynting-Robertson effect*
  default = 1.0
- `Solar Wind factor`, used in the Poynting-Robertson effect scheme 1.
  Ratio of solar wind drag to Poynting-Robertson drag.
  See *Solar Wind*
  default = 0.0
- `Asteroid emissivity eps`
  Thermal emissivity factor ϵ, used in the Yarkovsky effect.
  See *Yarkovsky effect*
  default = 0.95
- `Asteroid density rho`, in kg/ m^3
  Used in Yarkovsky effect, Poynting-Robertson effect and small bodies collision model
  See *Yarkovsky effect*, *Poynting-Robertson effect*, *Model for small bodies collisions*
  default = 3500.0 kg/ m^3
- `Asteroid specific heat capacity C`, in $\mathrm{J\,kg^{-1}K^{-1}}$
  Used in Yarkovsky effect
  See *Yarkovsky effect*
  default = 680 $\mathrm{J\,kg^{-1}K^{-1}}$
- `Asteroid albedo A`, Bond albedo, used for Yarkovsky effect.
  See *Yarkovsky effect*
  default = 0.2
- `Asteroid thermal conductivity K`, in $\mathrm{W\,m^{-1}K^{-1}}$
  Used for Yarkovsky effect
  See *Yarkovsky effect*
  default = 2.65 $\mathrm{W\,m^{-1}K^{-1}}$

2.6.8 Statistical small body collisions

- `Use Small Collisions`: Flag to enable model for small bodies collision model
  See *Model for small bodies collisions*
  - 0 (default): model is not enabled
  - 1: enable rotation reset model and fragmentation model for test particles.
  - 2: enable only rotation reset model for test particles.
  - 3: enable only fragmentation model for test particles.
- `Small Collisions Interval`, in time steps: Interval in which the small body collision model is called (default 1000).
  See *Model for small bodies collisions*
- `Nfragments`: Number of additional memory size for debris particles, in particle numbers, (default 0).
  See *Model for small bodies collisions* or *Particles Creation Model*

2.6.9 Create particles during integration

- `Create Particles file name`: file name for the particle creation model.
  See *Particles Creation Model*
  - -: no file, particle creation model is not used (default)
  - < particle creation file name>: This file is used to generate new particles during a simulation.
- `Asteroid collisional velocity V`, in m/s, used for small bodies collision model
  See *Model for small bodies collisions*
  default = 5000 m/s
- `Asteroid minimal fragment radius`, in m , used for small bodies collision model
  See *Model for small bodies collisions*
  default = 0.01
- `Asteroid fragment remove radius`, in m , used for small bodies collision model
  See *Model for small bodies collisions*
  default = 0.01 m
- `Nfragments`: Number of additional memory size for debris particles, in particle numbers, (default 0).
  See *Model for small bodies collisions* or *Particles Creation Model*

2.6.10 Options for encounters

- `Report Encounters`, flag to enable encounter information (default = 0).
  See *Report Encounters*.
  - 0: nothing happens.
  - 1: Encounter events between two bodies, with a separation less than `Report Encounters Radius` times the sum of their radii, are reported in the encounters-file. Encounters between test particles and other test particles are not reported.
  - 2: Report also encounters events between test particles and other test particles. This mode is only allowed when the test particle mode is used. Since the distance between test particles and other test particles is not calculated in the gravity calculation step, this mode needs another function call in order to find the encounters. See *Finding close encounter candidates between test particles*.
- `Report Encounters Radius`, used for `Report Encounters`, (default = 1.0).

In units of physical radii.
See *Report Encounters*.

- `Report Encounters Cloud Size`, used filter `Report Encounters` depending on the particle indices.
  See *Report Encounters*.
  - 1 (default): Encounter events between all particles are reported.
  - >1: Encounter events between particles belonging to the same particle cloud are not reported.
- `Stop at Encounter`, flag to stop simulations when a close encounter between two bodies happens, (default = 0).
  See *Stop at Encounter*.
  - 0: nothing happens.
  - 1: Simulations are stopped when the separation between two bodies is less than `Stop at Encounter Radius` times the Hill radius.
- `Stop at Encounter Radius`, used for `Stop at Encounter`, (default = 1.0).
  In units of Hill radii.

2.6.11 Options for collisions

- `Collision Precision`, in units of a physical radius fraction. (default 1.0^{-4})
  This parameter sets the tolerance of the detected collision time. See *Collision precision*.
  |`Collision Precision`| can not be smaller than 1.0^{-10}.
  - $precision$ > 0: particles overlap slightly, $r_{ij} < Ri + Rj$, $r_{ij} > (Ri + Rj) \cdot (1 - precision)$
  - $precision$ < 0: particles do not overlap, $r_{ij} > Ri + Rj$, $r_{ij} < (Ri + Rj) \cdot (1 + precision)$
- `Collision Time Shift`, in units of a physical radius factor (default 1.0).
  Allows to backtrace collision at a point before the collision, when the bodies are separated by an increased physical radius.
  See *Backtrace Collisions (TShift)*
- `Stop at Collision`, flag to stop simulations at the first collision time (default 0).
  This option is not supported in the multi simulation mode.
  See *Stop at Collision*
  - 0: nothing happens.
  - 1: stop simulation at the first collision time.
- `Stop Minimum Mass`, used in *Stop at Collision*, (default 0.0)
  Simulations are only stopped when **both** bodies have a mass larger than this value.
- `Collision Model`, can be used to implement a different collision model
  The default (0) is used for a perfect merger collision.
  See *Collisions*

2.6.12 Other

- `Set Elements file name`: file name for the set-element table.
  See *Set-elements function*.
  - -: no file, set-elements function is not used (default)
  - < data table file name>: This file is used to read the set-elements data table.

- `FormatS`: Output file format for multi simulation run.
  - 0: all simulations write to different files in their sub simulation directories (default).
  - 1: all simulations write to the same file in the master directory.
- `FormatT`: Output file format for time steps.
  - 0: all time steps are written to different files (default).
  - 1: all time steps are written to the same file.
- `FormatP`: Output file format for particles.
  - 0: all particles are written to different files.
  - 1: all particles are written to the same file (default).
- `FormatO`: Output file format for file names.
  - 0: file names contain time steps (default).
  - 1: file names contain output steps.
- `Serial Grouping`: Flag for exact reproducible results.
  See *Exact reproducible results*
  - 0: nothing happens.
  - 1: enable sorting step for exact reproducible results.
- `Do kernel tuning`: Flag to enable the self tuning routing of GPU kernel parameters.
  (default = 1)
  See *Use self tuning kernel parameters*
  - 0: if *The tuningParameters.dat file* is available, then read kernel parameters from that file.
  - 0: if *The tuningParameters.dat file* is not available, then use default values.
  - 1: run kernel tuning at the beginning of the integration and write parameters to *The tuningParameters.dat file*.
- `Do Kick in single precision`: Flag for precision in the kick force calculation
  See: *Use single precision in kick forces*
  - 0: use double precision in force terms (default)
  - 1: use single precision in force terms

2.6.13 Options for TTVs

- TTV file name = -
- RV file name = -
- TTV steps = 1
- Print Transits = 0
- Print RV = 0
- Print MCMC = 0
- MCMC NE = 0
- MCMC Restart = 0

2.7 Console Arguments

Instead of using the parameter file *param.dat*, some arguments can also be passed as console arguments. The console arguments have the highest priority and are overwriting the arguments of the *param.dat* file. The options are:

- -dt <f>: Time step in days
- -ei <i>: Energy output interval
- -ci <i>: Coordinates output interval
- -I <i>: Number of integration steps
- -n1 <f>: Value of n1
- -n2 <f>: Value of n2
- -ndev <i>: Number of devices to use
- -dev <i>: Device number
- -in <s>: Input file name
- -out <s>: Output name
- -R <i>: Restart a simulation at time step i
- -TP <i>: Test Particle mode i = 0, treat all bodies the same way, i = 1: small bodies don't affect big bodies.
- -M <s>: name of file, containing a list of the directories for the multi simulation mode.
- -Nmin <i>: Minimal number of bodies in the simulation, not including test particles.
- -NminTP <i>: Minimal number of test particles in the simulation.
- -SIO <i>: Order of symplectic integrator, The options are 2, 4 or 6.
- -aeN <s>: Name of the aeCount grid
- -t <f>: Start time of the simulation in years
- -MT <i>: Number of simulations in TTV calculations
- -sl <i>: Symplectic recursion levels
- -sls <i>: Symplectic recursion sub steps
- -collPrec <f>: Collision Precision
- -collTshift <f>: Collision Time Shift
- -GR <i>: Use GR
- -Nomp <i>: Number of parallel CPU cores to use. Only available in CPU version of GENGA

Here i means an integer, f a floating point value, and s a string.

2.8 Parameters in the define.h File

This file contains all default values of the parameters in the *param.dat* file. It also contains constants and additional parameters for GENGA. When a value is changed in this file, then GENGA needs to be recompiled.

2.8.1 Use old Shuffle

CUDA has replaced the shuffle functions `__shfl_xor` to a new function `__shfl_xor_sync`. GENGA offers a flag to switch between the two versions. If an old CUDA version is used (< CUDA 9.0) then the `def_OldShuffle` parameter in the `define.h` file must be set to 1.

- 0: use new sync shuffle version (default)
- 1: use old shuffle version

2.8.2 Ignore Lock File

- `def_IgnoreLockFile`, flag to set the behaviour of starting (not restarting) a new simulation - 0: When previous output files exist, then GENGA can only be started again after deleting the `lock.dat` file. This option prevents that existing data is overwritten. - 1: GENGA can be started even if previous files exist. Previous files are deleted and overwritten.

2.8.3 Memory options

- `def_MaxColl` (integer): Maximum number of collisions per time step that can be stored.
  When a collision happens, then the details of the collisions are stored in an internal buffer on the GPU. After the time, this buffer is transferred to the CPU and the information written into the collision file. The size of this buffer is specified by `def_MaxColl`. When more collisions occur in the same time step, then the integration is stopped and an error message written.
  Increasing this number also increases the total amount of needed memory.
  (default = 120)
- `def_SLevelsMax` (integer): Define the maximum number of symplectic substep levels.
  Increasing this number will increase the amount of needed memory.
  See *Higher level changeover functions*
- `def_NSetElementsMax` (integer): Define the maximum number of lines in the Set-Elements file.
  See *Set-elements function*.

2.8.4 Other

- `def_tol`: Tolerance in Bulirsch Stoer integrator (default = 1.0e-12).
- `def_dtmin`: Minimal time step in Bulirsch Stoer integrator (default = 1.0e-17).
- `def_NFileNameDigits`: set number of digits in the names of the output files (default = 12).
- `def_pc`: Factor in Pre-checker, pairs with $r_{ij}^2 < \text{pc}\, r_{\text{crit}}^2$ are considered as close encounter candidates.
  (default = 3.0)
  See *Finding close encounter candidates*.
- `def_pcf`: Single precision version of `def_pc`, used when `Do Kick in single precision = 1`.
  (default = 3.0f)
  See *Finding close encounter candidates* and *Use single precision in kick forces*.
- `def_cef`: Factor in close encounter detector, pairs with $r_{ij}^2 < \text{cef}\, r_{\text{crit}}^2$ are considered as close encounter pairs.
  (default = 1.0)
  See *Finding close encounter candidates*.
- `def_GMax`: Defines the maximum size of close encounter groups as 2^{GMax}.
  (default = 20)
- `def_poincareFlag`: Flag to enable the Poincare surface of section calculation

See *Poincare surface of section*

- 0: Poincare surface of section is not calculated
- 1: Poincare surface of section is calculated

- `def_Rand_Min`: lower limit for acceptable random number range (default = 1.0e-12).
  See *Model for small bodies collisions*.

2.8.5 Physical constants

The following physical constants are used by GENGA and can be changed in this section of the `define.h` file:

- `def_ksq = 1.0`
  Squared Gaussian gravitational constant in current units
- `def_Kg 2.959122082855911e-4`
  Squared Gaussian gravitational constant in $AU^3 day^{-2} M_{\odot}^{-1}$, used for conversion
- `dayUnit 0.01720209895`
- `def_AU 149597870700.0`
  AU in m
- `def_Solarmass 1.98855e30`
  Solar Mass in kg
- `def_c 299792458.0`
  speed of light in m/s
- `def_cm 10065.3201686`
  speed of light in AU / day * 0.0172020989
- `def_sigma 5.670373e-8`
  Stefan Boltzmann constant in $Jm^{-2}s^{-}1K^{-}4$

2.8.6 Parameters for the gas disk

See *Gas disk*

- `def_Gasnz_g`: Number of cells in z direction for gas grid
- `def_Gasnz_p`: Number of cells in z direction for particle grid
- `def_h_1`: scale height at 1AU for c = 1km/s
- `def_M_Enhance`: factor for enhancement
- `def_Mass_pl`: factor for enhancement
- `def_fMass_min`: factor for enhancement
- `def_Gas_cd`: numerical gas drag coefficient

CHAPTER

THREE

INTEGRATOR DETAILS

3.1 Integrator

GENGA is using a hybrid symplectic integrator [Chambers99].

The hybrid symplectic method uses a smooth changeover function to transfer the calculation of close encounters from the symplectic to a direct N-body integrator like the Bulirsch-Stoer method or something similar. This transition must be applied smoothly enough to prevent from too large energy errors. Therefore a critical radius must be defined to set a threshold between the close encounter phase and the normal integration phase, and it must be chosen large enough to ensure a smooth transition.

By using democratic coordinates, the Hamiltonian of a planetary system can be split into three parts:

$$H = H_A + H_B + H_C,$$

with

$$H_A = \sum_{i=1}^{N} \left(\frac{p_i^2}{2m_i} - \frac{Gm_i m_\star}{r_{i\star}} \right) - \sum_{i=1}^{N} \sum_{j=i+1}^{N} \frac{Gm_i m_j}{r_{ij}} [1 - K(r_{ij})],$$

$$H_B = - \sum_{i=1}^{N} \sum_{j=i+1}^{N} \frac{Gm_i m_j}{r_{ij}} K(r_{ij})$$

and

$$H_C = \frac{1}{2m_\star} \left(\sum_{i=1}^{N} \mathbf{p}_i \right)^2,$$

where the symbol $\star$ refers to the central mass, and $K(r_{ij})$ is a smooth changeover function ranging from 0 to 1. The tree parts H_A, H_B and H_C correspond to the Keplerian part, the interaction part and the Sun part of the Hamiltonian, respectively.

The limit where the changeover functions is applied is defined by a critical radius r_{crit} of a particle i:

$$r_{\text{crit},i} = \max(n1 \cdot R_{H,i}, n2 \cdot dt \cdot v_i). \tag{3.1}$$

It depends on two terms, the first contains the Hill radius R_H, the second contains the time step dt and the velocity v of the particle i. The two parameters $n1$ and $n2$ are typically set to 3 and 0.4.

The integrator needs to search for close encounter pairs at each time step and to sort them into independent close encounter groups. These groups are then integrated with the Bulirsch-Stoer direct N-body method. Ideally, the close

encounter groups consist of only a single pair of bodies, but it can happen that bodies have multiple close encounter pairs, which need to be linked together in a bigger close encounter group. In the worst scenario, all bodies are in a close encounter with some neighboring bodies, and all of them are linked together into a single giant close encounter group. This scenario is likely to happen, when the particle number density is increased for high resolution scenarios.

3.1.1 The n1 and n2 values

The values `n1` and `n2` from equation (3.1) can be set in the *param.dat* file. Typical values are n1 = 3.0 and n2 = 0.4.

3.1.2 The order of the symplectic integrator

The order of the symplectic integrator can be set with the `Order of integrator` parameter in the *param.dat* file. Options are 2, 4 or 6.

The 4th and 6th order symplectic integrators use the description of [Yoshida90].

The higher order integrators work the best for cases with few close encounters.

3.1.3 Finding close encounter candidates

During the force calculation, the distance of all pairs of bodies are calculated. During this step, close encounter candidates are reported to a list when the mutual distance is smaller then the critical radius:

$$r_{ij}^2 < \text{pc}\, r_{\text{crit}}^2.$$

The factor `pc` is a safety factor. It can be set in the *define.h* file (default = 3.0).

After all close encounter candidates are found. The real minimal distance between the particles is calculated, by interpolating between the time steps. Close encounters are reported when:

$$r_{ij,min}^2 < \text{cef}\, r_{\text{crit}}^2.$$

The factor `cef` is a safety factor. It can be set in the *define.h* file (default = 1.0).

3.1.4 Finding close encounter candidates between test particles

When encounter events between test particles and other test particles should be reported, then another function call is needed to find them, because the distance between test particles is not calculated in the gravitational force evaluation. When the number of test particles is $\lessapprox 10'000$ then an N^2 algorithm can be used, where each test particle checks its distance to every other test particle and reports close encounter candidates. But when the number of test particles gets larger, then this approach gets very inefficient. In that case a Bounding Volume Hierarchy (BVH) tree method is used to check all potential close encounter candidates.

3.1.5 Close Encounters Pairs

The `Maximum encounter pairs` parameter in the *param.dat* file, sets the amount of memory that is allocated to store close encounter pairs of each body. When a body has more close encounters that specified here, then the simulation is stopped and an error message is written. Setting a larger value of `Maximum encounter pairs` increases the memory usage of the code.

3.1.6 Higher level changeover functions

The `Symplectic recursion levels` and the `Symplectic recursion sub steps` parameters in the *param.dat* file can be uses to enable higher order changeover functions. `Symplectic recursion levels = 1` corresponds to the original hybrid symplectic integration method, where forces between close encounter pairs get smoothly moved from the symplectic integrator into the direct Bulirsch-Stoer method. The higher level changeover functions define

more levels in between. In the second level, the time step gets reduced in into `Symplectic recursion sub steps` sub steps but still integrated with a symplectic integrator. The highest level is then the Bulirsch-Stoer integrator.

Since the higher level changeover function requires more memory to store the close encounter pairs, not more than `Symplectic recursion levels = 3` should be used. However if more levels than three are needed, then the `def_SLevelsMax` parameter in the *define.h* file must be adjusted. Typical values for `Symplectic recursion sub steps` are 2,4,8 or 10. A good strategy to set an optimal choice of parameter is to increase the number of levels and sub steps until no more than ~ 512 close encounter pairs are reported in the info file.

By using the option `Symplectic recursion levels = -1`, a self tuning routine is called before the start of the integration, to find the fastest option by itself.

In Figure Fig. 3.1 are shown some examples of higher order changeover functions.

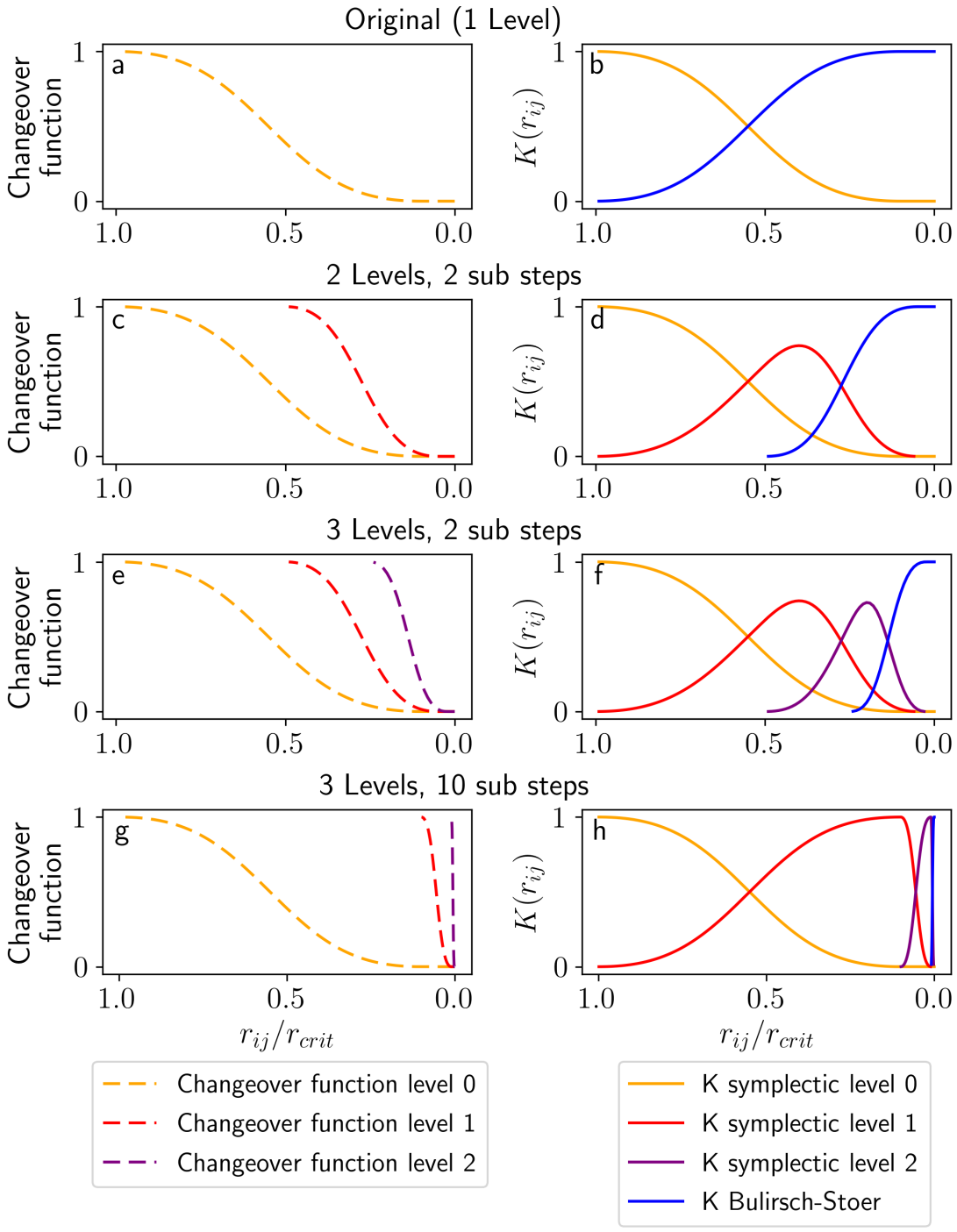

Fig. 3.1: Panels a and b: Original hybrid symplectic method: a changeover function switches smoothly from K symplectic to K Bulirsch-Stoer. Panels c and d: A second changeover function is included. The basic symplectic step is divided in the first level into two sub steps. The second level is the Bulirsch-Stoer method. Panels e and f: Two additional changeover functions are included for a three level scheme with each two sub steps. Panels g and h: Two additional changeover functions are included for a three level scheme with each ten sub steps.

CHAPTER

FOUR

OUTPUT FILES

4.1 Output Files

4.1.1 The Lock File: lock.dat

This file is created at the beginning of a GENGA simulation. The behaviour depends on the `IgnoreLockFile` in the *define.h* file.

- If it is set to 0, then GENGA can not be started again from time step 0.
  The lock file prevents that output files are overwritten and lost.
  GENGA can only be started again when the lock.dat file is deleted.
- If it is set to 1, then GENGA can always be started again from time step 0, and all output files are overwritten.

Restarting GENGA from a time step > 0 is not affected by the lock file.

4.1.2 The Master File: master.out

The master file contains information about the used hardware and simulation progress. If an error occurs then the master file contains more details about that. The master file is not deleted at a new GENGA start.

4.1.3 The information file: info<name>.dat

This file contains general information about the used parameters and hardware.

At the beginning, the file lists all used parameters, the version number and driver information.

After the parameters, the file lists the timings of the kernel tuning routine (see *Use self tuning kernel parameters*).

At each energy output interval the file gives information of the number of close encounter.

- Precheck-pairs: number of close encounter candidates found from the prechecker (see *Finding close encounter candidates*).
- CE: total number of detected close encounter pairs (see *Finding close encounter candidates*).
- groups: number of separate close encounter groups; followed by the number of close encounter groups of size 2,4,8,16,32,64,128,256,512,1024,2048 ...

If an error occurs during the simulation, then in this File is written the last coordinates-Output and an information about the error.

4.1.4 The tuningParameters.dat file

See *Use self tuning kernel parameters*.

If the kernel self tuning is enabled, then this file is created, containing the values of the kernel parameters. If the kernel self tuning is disabled, then kernel parameters can be read from this file. The later option can be useful for performance measurement.

4.1.5 The Coordinate Output Files: Out<name>.dat

This file contains the heliocentric positions and velocities, the spin and some information about the orbit and close encounters. At the `coordinate output interval`, set in the *param.dat* file, a new output is written.

The output file format can be set to text-format or to binary-format with the `Use output binary files` in the *param.dat* file.

The number of digits in the output file names can be changed with the `def_NFileNameDigits` parameter in the *define.h* file.

Output File Format

The values of the output files depend on the `Output file Format`, set in the *param.dat* file.
Possible options are: (default = t i m r x y z vx vy vz Sx Sy Sz amin amax emin emax aec aecT encc test)

- t: time, in years.
- i: index of the body.
- m: mass in Solar masses.
- r: physical radius in AU.
- x: x-position in AU (heliocentric).
- y: y-position in AU (heliocentric).
- z: z-position in AU (heliocentric).
- vx: x-velocity in AU/day * 0.0172020989 (heliocentric) (See *Units*).
- vy: y-velocity in AU/day * 0.0172020989 (heliocentric) (See *Units*).
- vz: z-velocity in AU/day * 0.0172020989 (heliocentric) (See *Units*).
- Sx: x-spin in Solar masses AU^2 / day * 0.0172020989. (See *Units*).
- Sy: y-spin in Solar masses AU^2 / day * 0.0172020989. (See *Units*).
- Sz: z-spin in Solar masses AU^2 / day * 0.0172020989. (See *Units*).
- amin: minimal value of semi major axis range for aecount (See *aeLimits*).
  Optional.
- amax: maximal value of semi major axis range for aecount (See *aeLimits*).
  Optional.
- emin: minimal value of eccentricity range for aecount (See *aeLimits*).
  Optional.
- emax: maximal value of eccentricity range for aecount (See *aeLimits*).
  Optional.
- k2: potential Love number of degree 2, dimensionless.

(needed when given as initial conditions)

- k2f: fluid Love number of degree 2, dimensionless.
  (needed when given as initial conditions)
- tau: time lag in day / 0.0172020989 (See *Units*).
  (needed when given as initial conditions)
- Ic: moment of inertia, dimensionless (See *Units*).
  (needed when given as initial conditions)
- aec: aecount is the number of time steps since the last coordinate output time in which the particles semi major axis and eccentricity where in the aecount box limits (see *aeLimits*).
  Optional.
- aecT, aecountT is the integrated value of all previous aecount values.
  Optional.
- encc: enccountT is the number of time steps since the simulation start in which the particle was in a close encounter with another (massive) particle.
  Optional.
- Rc: Critical radius for close encounters, rcrit, in AU
  Optional.
- test: value stored in the test arrays.
  Optional.

When the `Use output binary files` option is used, then the files contain the same data as described before with the following types:

- t: double, 64bit
- i: int, 32bit
- m: double, 64bit
- r: double, 64bit
- x: double, 64bit
- y: double, 64bit
- z: double, 64bit
- vx: double, 64bit
- vy: double, 64bit
- vz: double, 64bit
- Sx: double, 64bit
- Sy: double, 64bit
- Sz: double, 64bit
- amin: float, 32bit
- amax: float, 32bit
- emin: float, 32bit
- emax: float, 32bit
- k2: double, 64bit

- k2f: double, 64bit
- tau: double, 64bit
- Ic: double, 64bit
- aec: float, 32bit
- aecT: float, 32bit
- encc: unsigned long long, 64bit
- Rc: double, 64bit
- test: double, 64bit

The structure of the coordinate output files depends on the parameters `FormatS`, `FormatT`, `FormatP` and `FormatO`. Here we describe the possible choices:

The tools *Convert_P1T0_to_P0T1* and *Convert_P0T1_to_P1T0* can be used to convert output files between different formats.

FormatS = 0, FormatT = 0, FormatP = 1, FormatO = 0: Out<name>_<time step>.dat

In this format, at each coordinate output interval, a new file is created which contains all particles. In the multi simulation mode, each sub simulation folder contain individual files:

```
t i1 m1 r1 x1 y1 z1 vx1 vy1 vz1 Sx1 Sy2 Sz1 ...
t i2 m2 r2 x2 y2 z2 vx2 vy2 vz2 Sx2 Sy2 Sz2 ...
.
.
.
t in mn rn xn yn zn vxn vyn vzn Sxn Syn Szn ...
```

FormatS = 1, FormatT = 0 FormatP = 1, FormatO = 0: Out<name>.dat

Here the difference is that in the multi simulation mode, the coordinates are not written in the sub simulation folders, but in the main folder. The output files contain all particles from all sub simulations.

FormatS = 0, FormatT = 1 FormatP = 1, FormatO = 0: Out<name>.dat

Here all the time steps are written to the same file, containing all time steps and all particles.

FormatS = 1, FormatT = 1 FormatP = 1, FormatO = 0: Out<name>.dat

Here all time steps are written to the same file, containing all time steps and all particles from all sub simulations.

FormatP = 0: Outp< index>.dat

Here all particles are written to different files, containing all time steps.

FormatS = 0, FormatT = 0, FormatP = 1, FormatO = 1: Out<name>_<output step>.dat

Here the difference is that the output files are not named after the time step, but the output step. When the simulation is interrupted at a time step in between of two output steps, then a backup file 'Outbackup<name>_<time step>.dat' is created. This backup step can be read by the restart option -R -1. To restart from a normal output file, the real time step, and not the output step must be chosen.

FormatT = 0, and FormatP = 0

This option is not possible, it is equivalent to FormatT = 1 and FormatP = 0

FormatS = 1, FormatT = 1, and FormatP = 0

This option is not possible, it is equivalent to FormatS = 0, FormatT = 1 and FormatP = 0

Number of outputs per interval

When this number is larger than 1, then at each coordinate output interval, n consecutive outputs are written.

For example, when the following numbers are set

- `Coordinates output interval` = 100
- `Number of outputs per interval` = 5,

then the following time steps are written as outputs: 0, 96, 97, 98, 99, 100, 196, 197, 198, 199, 200, ...

4.1.6 The Irregular Coordinate Output Files: OutIrr<name><time>.dat

These files are only created when the output calendar file is used. The file contains the same structure as the *The Coordinate Output Files: Out<name>.dat* but at the time specified in the calendar file. The number in the file name corresponds to the line in the calendar file. See *Irregular output times*.

The number of digits in the output file names can be changed with the `def_NFileNameDigits` parameter in the *define.h* file.

4.1.7 Keplerian-Elemets Output Files: aei<name>.dat

The Coordinate output files can be uses to generate Keplerian-Elements output files with the *KE* tool. These files contain the following:

```
time i a e inc Omega w Theta E M m r
```

with

- time: time, in years
- i: index
- a: semi major axis, in AU
- e: eccentricity:
- inc: inclination, in radians
- Omega: longitude of the ascending node, in radians
- w: argument of periapsis, in radians
- Theta: true anomaly, in radians
- E: eccentric anomaly, in radians
- M: mean anomaly, in radians
- m: mass, in Solar masses
- r: radius, in AU

4.1.8 Barycentric output Files: OutBary<name>.dat

The barycentric coordinate output files can be generated with the *ConvertHelioToBarry* tool.

The files contain the same information as the original heliocentric coordinate output files, but they contain at the beginning an additional particles, representing the barycentrum, with an index of -1.

4.1.9 The Energy Output File: Energy<name>.dat

This file contains information about the number of particles, angular momentum and the energy. At the `Energy output interval`, set in the *param.dat* file, a new line in this file is written. The format is the following:

```
time0  N  V  T  LI  U  ETotal  LTotal  LRelativ  ERelativ
time1  N  V  T  LI  U  ETotal  LTotal  LRelativ  ERelativ
.
.
.
```

with

- time in years
- N: Number of particles
- V: Total potential energy , in $M_\odot AU^2/day^2$
- T: Total Kinetic energy, in $M_\odot AU^2/day^2$
- LI: Angular momentum lost at ejections, in $M_\odot AU^2/day$
- U: Inner energy created from collisions, ejections or gas disk, $M_\odot AU^2/day^2$
- ETotal: Total Energy, in $M_\odot AU^2/day^2$
- LTotal: Total Angular Momentum, in $M_\odot AU^2/day$
- LRelativ: (LTotal_t - LTotal_0)/LTotal_0, dimensionless
- ERelativ: (ETotal_t - ETotal_0)/ETotal_0, dimensionless

The tool `cleanEnergy.py` in the `tools` directory can be used to clean the Energy files from restarting data.

See *Cleaning the Energy files*.

4.1.10 The Irregular Energy Output File: EnergyIrr<name>.dat

See *Irregular output times*.

This file is only created when the output calendar file is used. The file contains the same structure as the regular *The Energy Output File: Energy<name>.dat*, but at output times, specified in the output calendar file.

4.1.11 The Execution Time File: time<name>.dat

This file contains the execution time spent for the corresponding Coordinate Output interval, in seconds. The last line contains the total execution time in seconds. The first column indicates the time step. This last entry in the file is used for the automated restart (restart timestep = -1).

When the code was interrupted in between of two output intervals, then the time file contains additional lines with the divided times in it. If the code was interrupted many times due to e.g. limited wall times, the time files can get hard to read.

The tool `cleanTime.py` in the `tools` directory can be used to clean the time files and add the divided times together to the original coordinate output intervals.

See *Cleaning the time files*.

4.1.12 The Collisions File: Collisions<name>.dat

See *Collisions*.

In this file are listed the details of the collisions between particle i and j. The precision of the collision output can be adjusted with the `Collision Precision` argument in the *param.dat* file (See *Collision precision*). The file contains the following columns:

```
time indexi mi ri xi yi zi vxi vyi vzi Sxi Syi Szi indexj mj rj xj yj zj vxj vyj vzj Sxj␣
↪Syj Szj
.
.
.
```

4.1.13 The Tshift Collisions File: CollisionsTshift<name>.dat

See *Backtrace Collisions (TShift)*.

In this file are listed the details of the backtraced collisions between particle i and j. The collision time shift option can be set by the `Collision Time Shift` argument in the *param.dat* file. This file is only created when `Collision Time Shift` is used. The file contains the following columns:

```
time indexi mi ri xi yi zi vxi vyi vzi Sxi Syi Szi indexj mj rj xj yj zj vxj vyj vzj Sxj␣
↪Syj Szj
.
.
.
```

4.1.14 The Stop-at-collision-file: OutCollision.dat

See *Stop at Collision*.

This file is only created when the `Stop at Collision` option is enabled. It contains all particles of the simulation at the time when the first collision occurred. The file contains the same columns as the normal output files.

4.1.15 The encounter-file: Encounters<name>.dat

See *Report Encounters*.

This file is only created when the `Report Encounters` option is enabled. It contains the details of each encounter event:

```
time indexi mi ri xi yi zi vxi vyi vzi Sxi Syi Szi indexj mj rj xj yj zj vxj vyj vzj Sxj␣
↪Syj Sz
.
.
.
```

4.1.16 The ejection file: Ejections<name>.dat

See *Ejections*.

This file contains the details of all ejection events, in the format:

```
time index m r x y z vx vy vz Sx Sy Sz case
.
.
.
```

with: case = -3 for bodies removed at the outer boundary, and case = -2 for bodies removed at the inner boundary.

4.1.17 The stellar evolution file: Star<name>.dat

See *Tidal Forces*.

This file is only produced when `Use Tides` or `Use Rotational Deformation` are enabled. The file contains the parameters of the star in the format:

```
time mass radius Spin_x Spin_y Spin_z Ic Love-number fluid-Love-number time-lag
.
.
.
```

4.1.18 The Irregular stellar evolution file: StarIrr<name>.dat

See *Tidal Forces*.

This file is only created when the output calendar file is used. The file contains the same structure as the regular *The stellar evolution file: Star<name>.dat*, but at output times, specified in the output calendar file.

4.1.19 The Fragments File: Fragments<name>.dat

See *Model for small bodies collisions*.

This file is only created when the model for small bodies collisions `UseSmallCollisions` or the particle creation mode `Create Particles` in the *param.dat* file are enabled. The file contains information about fragmentation and rotation reset events:

```
time index m r x y z vx vy vz Sx Sy Sz event
.
.
.
```

the 'event' indicates the following:

- **0: | rotation rate reset**

  Used in `UseSmallCollisions` = 1 or 2 mode.

- **-1: | Collision, the particle is destroyed, and it is replaced with new fragments (listed in the next lines with event=1 or event=2 of this file)**

  Used in `UseSmallCollisions` = 1 or 3 mode.

- **1: | A new fragment particle. The original body is the last body in this file with event = -1.**

  Used in `UseSmallCollisions` = 1 or 3 mode.

- **2: | A new fragment particle. The original body is the last body in this file with event = -1.**

  This body is too small and it is directly removed from the simulation.
  Used in `UseSmallCollisions` = 1 or 3 mode.

- **10: | A new particle is created.**

Used in `Create Particles` = 1 mode.

Each newly created fragment gets a new, increasing, index number. This file permits to reconstruct the collision and fragmentation history of every particle.

4.1.20 The a-e and a-i grid files: aeCount<grid name><time step>.dat

See *The a-e and a-i grid*.

These files are only written when the aegrid option is used. The files contain four matrices, separated by a blank line.

- a-e counts since the last coordinate output, size Na x Ne
- a-e counts since the beginning of the simulation, size Na x Ne
- a-i counts since the last coordinate output, size Na x Ni
- a-i counts since the beginning of the simulation, size Na x Ni

4.1.21 The Poincare surface of section file: Poincare<name><timeInterval>.dat

See *Poincare surface of section*

The file contains the coordinates of the Poincare surface of section:

```
time index x vx
.
.
.
```

The crossing events are written consecutively to the file. After each coordinate output interval, another file is created to reduce the file sizes.

CHAPTER

FIVE

OPTIONS

5.1 Collisions

A collision between two particles happens when the separation r_{ij} between two bodies i and j gets smaller than (or close to) the sum of their physical radii $R_i + R_j$. The current version of GENGA treats collisions as perfect inelastic mergers by forming one single bigger body. During this collision process, linear momentum is conserved. Physically, a part of the potential and kinetic energy is transformed into an internal energy. GENGA keeps track of this internal energy U, such that the overall energy of the system is conserved. Angular momentum is conserved by transferring the angular momentum of the two bodies into the spin of the new body.

The `Collision Model` option can be used to implement a different collision model than perfect merger. This can be done int the `collide` function in the file `directAcc.h`.

When a collision happens, the coordinates of the two involved bodies are reported in the collision file (see *The Collisions File: Collisions<name>.dat*).

The following parameters are relevant for the collision handling and can be set in the *param.dat* file:

- `Collision Precision` in units of a physical radius fraction. (default 1.0^{-4})
- `Collision Time Shift`, in units of a physical radius factor. (default 1.0)
- `Stop at Collision`, flag to stop simulations at the first collision time (default 0)
- `Stop Minimum Mass`, used for *Stop at Collision*, (default 0).
- `Collision Model`, can be used to implement a different collision model. The default (0) is used for a perfect merger collision.

Relevant parameters in the *define.h* file are:

- `def_MaxColl`

5.1.1 Collision details

The position and velocity of the new body is calculated as

$$\mathbf{x}_{\rm new} = \frac{\mathbf{x}_i m_i + \mathbf{x}_j m_j}{m_i + m_j}$$
$$\mathbf{v}_{\rm new} = \frac{\mathbf{v}_i m_i + \mathbf{v}_j m_j}{m_i + m_j}$$

The spin $\mathbf{S}$ of the new body is calculated as

$$\mathbf{L}_{ij} = \frac{m_i m_j}{m_i + m_j} (\mathbf{r}_{ij} \times \mathbf{v}_{ij})$$
$$\mathbf{S}_{\text{new}} = \mathbf{S}_i + \mathbf{S}_j + \mathbf{L}_{ij}$$

In order to keep track of the energy conservation, we add the lost kinetic and potential energy from collisions, ejections and caused by the gas drag into a quantity U. This quantity includes the energy loss from all particles together and not from single particles. U is not directly related to a physical quantity, however it contains the change of the inner energy of all particles and the spin energy caused at collision.

When two particles i and j collide, then U is increased by

$$U = \frac{1}{2} \frac{m_i m_j}{m_i + m_j} v_{ij}^2 - G \frac{m_i m_j}{r_{ij}}$$

The radius R of the new particle is set by conserving the mass and by mixing the densities of the two particles.

$$R_{\text{new}} = \left(R_i^3 + R_j^3\right)^{1/3}$$

The index of the next body is calculated according to the rules:

- The index of the more massive body.
- If both bodies have an equal mass, then take the smaller index of the bodies i and j

The moment of inertia Ic of the new body is calculated as

$$Ic_{\text{new}} = \frac{m_i Ic_i + m_j Ic_j}{m_i + m_j}$$

The potential love number $k2$ of the new body is calculated as

$$k2_{\text{new}} = \frac{m_i k2_i + m_j k2_j}{m_i + m_j}$$

The fluid love number $k2f$ of the new body is calculated as

$$k2f_{\text{new}} = \frac{m_i k2f_i + m_j k2f_j}{m_i + m_j}$$

The time lag τ of the new body is calculated as

$$\tau_{\text{new}} = \frac{m_i \tau_i + m_j \tau_j}{m_i + m_j}$$

At the end of the collision process, the body i is transferred to be the new body, and body j is marked as a ghost particle, which is then removed from the simulation later.

Collisions can happen only during a close encounter process, and are called during the Bulirsh-Stoer integration. The implementation of the collision can be found in the `collide` function in the `directAcc.h` file.

5.1.2 Collision precision

The collision process is resolved during the Bulirsh-Stoer direct integration with discrete time steps. Therefore, a collision is generally not detected at the exact collision time, but rather when the two particles already overlap by a small amount. Using `Collision Precision = 1.0`, GENGA uses the coordinate from the Bulirsh-Stoer step when the collision is first detected. This must be considered when using the data from the collision file or also directly within the code for further analysis.

Using `Collision Precision < 1.0`, GENGA refines the collision time to $\frac{(R_i+R_j)-r_{ij}}{R_i+R_j} < precision$, where r_{ij} the separation between the two bodies and R the physical radius. In this way the reported coordinates from the collision slightly overlap.

- $precision > 0$: particles overlap slightly, $r_{ij} < Ri + Rj$, $r_{ij} > (Ri + Rj) \cdot (1 - precision)$

Using a negative value for `Collision Precision < 1.0`, the reported coordinates at collision are not overlapping, the distance between the two particles is slightly larger then the sum of the two radii.

- $precision < 0$: particles do not overlap, $r_{ij} > Ri + Rj$, $r_{ij} < (Ri + Rj) \cdot (1 + precision)$

The precision should not be set smaller than 1.0^{-10}. When the exact time of a collision is important, a value of around 1^{-4} is recommended. Note that this parameter causes some more iterations in the Bulirsh-Stoer routine and can slightly increase the run time of a simulation.

In Fig. 5.1 is shown an example of the influence of the collision precision, and in Fig. 5.2 is shown a zoomed region with different `Collision Precision` values.

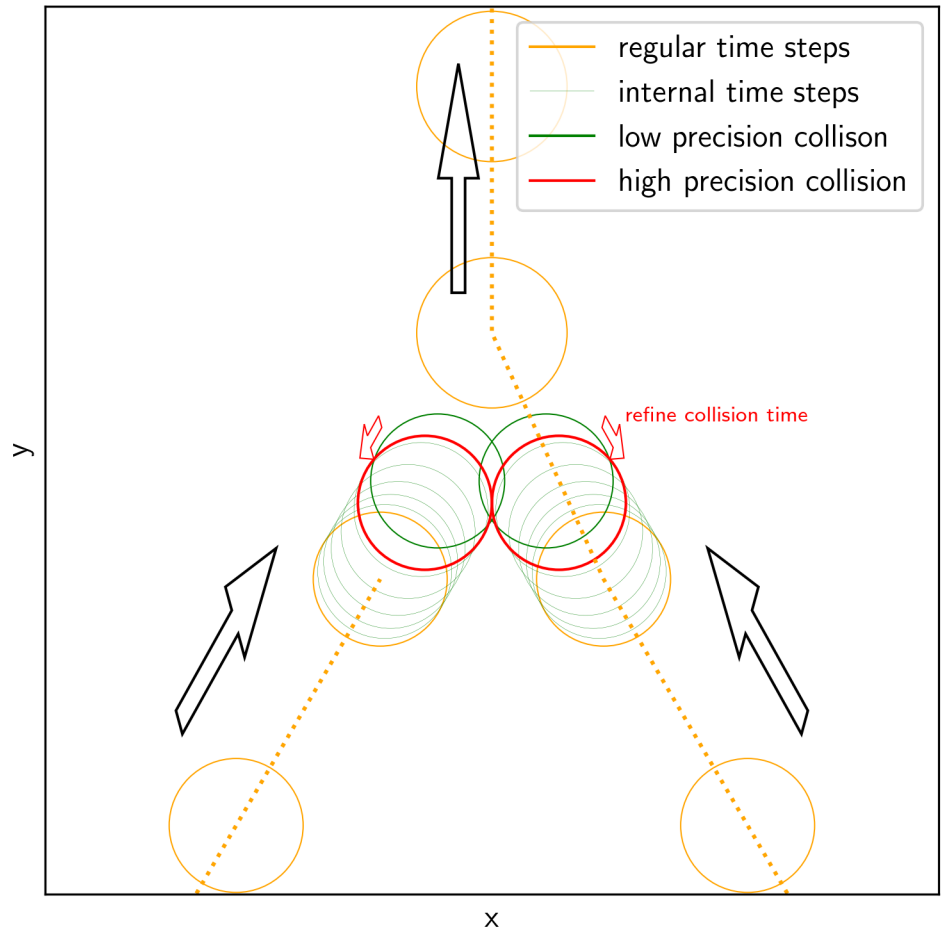

Fig. 5.1: Collision and merging of two bodies. In orange are shown the regular time steps before and after the collision. In thin green are shown the internal time steps of the Bulirsh-Stoer close encounter integration. A collision is reported when the two bodies already overlap (green color). By using a high collision precision, the collision located is resolved at the exact contact time (red color).

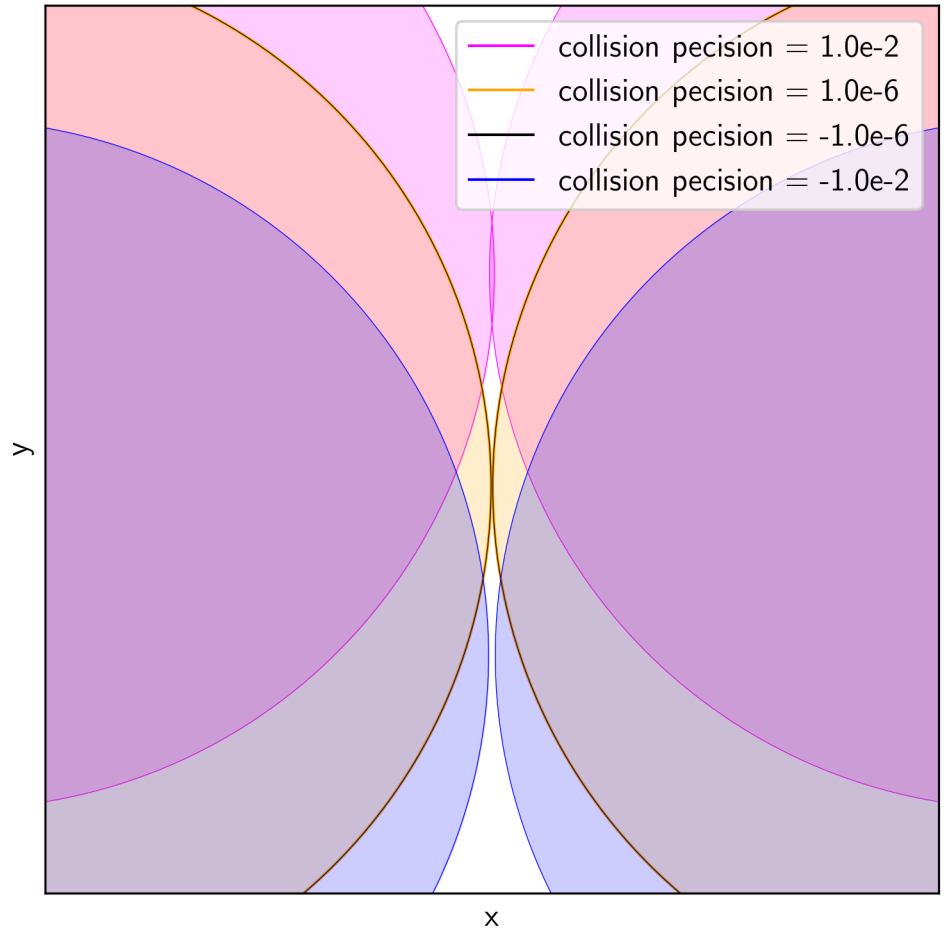

Fig. 5.2: Collision position of different `Collision Precision` values. A positive value leads to slightly overlapping bodies. With a negative value, the bodies do not overlap, and the collision is reported slightly before the bodies really touch.

5.1.3 Backtrace Collisions (TShift)

The `Collision Time Shift` argument allows to backtrace a detected collision to a time prior to the real collision time, when the two bodies were separated by a factor f times their physical radii. The factor f is set by `Collision Time Shift`. This option is especially useful when more complex collision models than perfect mergers are used. Backtraced collisions are only calculated when the involved bodies really will collide. If they will miss a collision and undergo a close flyby, then this option is not activated. This option can be combined with the `Collision Precision` option.

This option is only activated for bodies with a mass larger than `Stop Minimum Mass`.

Since multiple collisions can occur at a similar time, the backtrace option has to resolve every collision isolated. This can result in a longer run time of the code. Especially when many collisions occur.

Backtraced collisions are reported in the file *The Tshift Collisions File: CollisionsTshift<name>.dat*.

In Fig. 5.3 is shown an example of a backtraced collision.

5.1.4 Stop at Collision

The `Stop at Collision` option allows to stop a simulation at the time when the first collision occurs. With this option enabled, GENGA integrates all bodies in the simulation to the exact -or backtraced - collision time. Since the time of the first collision does not correspond in general with an output time, GENGA creates a separate output file with the coordinates of all particles at the time of the first collision. See *The Stop-at-collision-file: OutCollision.dat*. This option is useful when collisions between bodies are resolved with an external code. Then GENGA can be stopped when (or before) a collision happens, the collision resolved externally, and finally GENGA restarted again.

A simulation is only stopped at a collision when **both** involved particles have a mass larger than `Stop Minimum Mass` (default 0).

5.2 Encounters

This section describes the options or reporting encounter events. It is not related to the close encounter arguments used for the changeover function of the integrator.

The following parameters are relevant for the encounter-report or stop-at-encounters handling and can be set in the *param.dat* file:

- `Report Encounters`
- `Report Encounters Radius`
- `Stop at Encounter`
- `Stop at Encounter Radius`

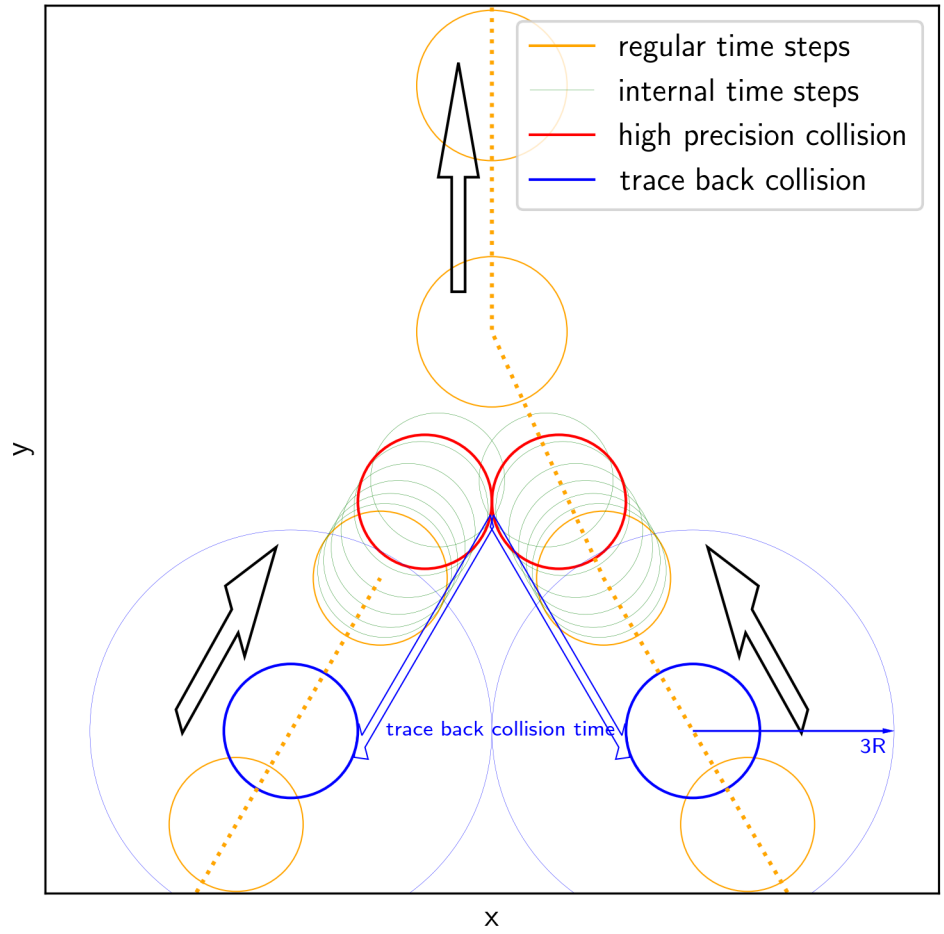

Fig. 5.3: The real collision (in red color) is backtraced until the time when the two bodies are separated by a factor f times the sum of their physical radii. In this example, we use $f = 3$. The location of the backtraced collision is shown is blue color.

5.2.1 Report Encounters

An encounter event is defined as the moment when the distance between two bodies has reached a minimum. Typically an encounter event happens only once per orbit. Further, only encounter events are considered with

$$r_{ij} < f * R_i + f * R_j,$$

where r_{if} is the separation between the two involved bodies, R is the physical radius and the factor f is the value given in `Report Encounters Radius`.

Note that the reported coordinates of the encounter events are not exactly refined to the true closest approach, instead the coordinates of the next Bulirsh-Stoer time step is reported.

When this closest approach happens within a close encounter integration phase, then it is directly detected by the interpolation polynomial used in *Close Encounters Pairs*. When the closest approach happens outside of a close encounter integration, meaning when the value of `Report Encounters Radius` time the sum of the two physical radii is larger than the mutual critical radius, then the critical radii are automatically increased by this value. As a consequence, the integration can be slowed down if `Report Encounters Radius` is set too large. We recommend a value between 1 and 100.

When a collision happens, the coordinates of the two involved bodies are reported in the encounters file (see *The encounter-file: Encounters<name>.dat*).

5.2.2 Report Encounters Cloud Size

A particle cloud is a collection of particles, representing the same physical object. Encounters between particles belonging to the same cloud are not reported in the encounters file. The cloud index is computed via the particles index and the cloud size as

$$\text{Cloud index} = (\text{round to integer}) \left(\frac{\text{Particle index}}{\text{Cloud Size}} \right),$$

where the cloud size is set by the `Report Encounters Cloud Size` parameter.

Therefore, the particle indices can be used to assign particles to different particle clouds. If the cloud size is equal to 1, then all encounters between all particles are reported.

5.2.3 Stop at Encounter

When this options is enabled, then simulations are stopped when a close encounter occur with

$$r_{ij} < g * RH_i + g * RH_j,$$

where r_{if} is the separation between the two involved bodies, RH is the Hill radius and the factor g is the value given in `Stop at Encounter Radius`.

5.3 Ejections

The following parameters are relevant for ejections and can be set in the *param.dat* file:

- `Inner truncation radius`
- `Outer truncation radius`

When the distance of a particle gets smaller than `Inner truncation radius`, or when it gets larger than `Outer truncation radius`, then the particle is removed from the simulation. The distance is calculated as:

$$r = \sqrt{x^2 + y^2 + z^2}$$

The energy of the removed particle is calculated and added to the inner energy term of the simulation. All ejection events are reported in the *The ejection file: Ejections<name>.dat*.

5.4 aeLimits

The aeLimits can be used to check how much time a specific body spends in a given range in semi-major axis and eccentricity. The limits of this range can be set with the `amin`, `amax`, `emin` and `emax` parameters in the initial condition file. When the corresponding particle spends time in the given range in a-e space, then a counter is increased. The values of the counter is included in *The Coordinate Output Files: Out<name>.dat*. These values can be useful in a stability analysis of planetary systems.

5.5 The a-e and a-i grid

The a-e (semi-major axis - eccentricity) and a-i (semi-major axis - inclination) grids can be used to keep track of the coordinates of bodies at each time step, also in between of the specified coordinate output intervals. These grids count at every time step the number of particles in each cell. This option offers to possibility to keep track of small scale movements of particles without having to write too many output files.

The dimensions and resolutions of the grid can be specified by the following user parameters:

- `Use aeGrid`, flag to enable the a-e and a-i grids, (default = 0)
- `aeGrid amin`, minimal value of the same-major axis dimension, in AU.
- `aeGrid amax`, maximal value of the same-major axis dimension, in AU.
- `aeGrid emin`, minimal value of the eccentricity dimension.
- `aeGrid emax`, maximal value of the eccentricity dimension.
- `aeGrid imin`, minimal value of the inclination dimension, in radians.
- `aeGrid imax`, maximal value of the inclination dimension, in radians.
- `aeGrid Na`, number of points in the same-major axis dimension.
- `aeGrid Ne`, number of points in the eccentricity dimension.
- `aeGrid Ni`, number of points in the inclination dimension.
- `aeGrid Start Count`, starting time step when a-e and a-i grid start, in units of time steps
- `aeGrid name`, name of the grid files.

The a-e and a-i grid consist both of two parts, the first grid counts the number of particles per cell since the last coordinate output time, and the second grid counts the overall number since the beginning of the simulations. At each coordinate output time, also the a-e and a-i grid data is transferred to the CPU and written to the files *The a-e and a-i grid files: aeCount<grid name><time step>.dat*.

When using the multi simulation mode, then all simulations contribute to the same grid.

In Fig. 5.4 is shown an example of an a-e grid with the following parameters:

```
Use aeGrid = 1
aeGrid amin = 0
aeGrid amax = 5
aeGrid emin = 0
aeGrid emax = 0.8
aeGrid imin = 0
aeGrid imax = 0.8
aeGrid Na = 400
aeGrid Ne = 400
aeGrid Ni = 400
```

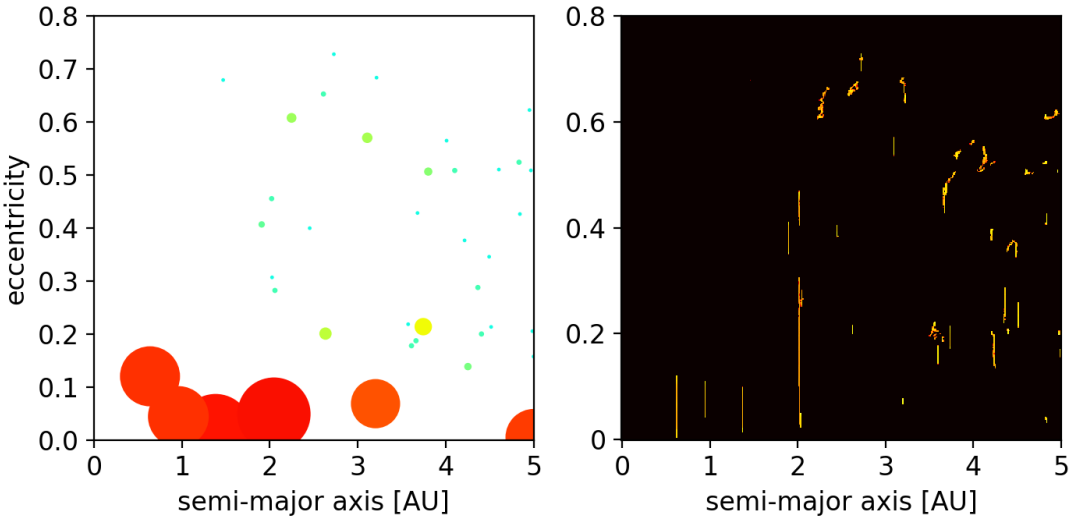

Fig. 5.4: Left panel: a snapshot of a planet formation simulation with six formed terrestrial planets (shown in red color) and some leftover planetesimals (shown in blue and green colors).
Right panel: The a-e grid of the same time step includes all positions since the last output time and visualizes the movement of individual bodies.

In Fig. 5.5 is shown another example of an a-e grid, by integrating the Solar System.

5.6 Set-elements function

(Attention: Update since version 3.92, units and data structure changed)

This option can be used to modify the orbital parameters of a body, according to a precomputed data table. The data table must be provided as a file and the name of the file must be specified with the `Set Elements file name` in the *param.dat* file. The data table can either include Keplerian elements or cartesian coordinates, but the two sets can not be mixed. If cartesian coordinates are used, then the positions must be given in heliocentric coordinates, and the velocities can either be given in barycentric or heliocentric coordinates. The Keplerian or cartesian coordinate sets must not be complete, they can also contain only a subset.

All elements are interpolated with a cubic interpolation scheme from the provided data table. The data table must provide elements for at least four different times. With less than four times, the cubic interpolation can not be done. The data table must also provide a time entry at or after the end of the simulation.

The length of the file can not be larger than the value of `def_NSetElementsMax` in *define.h* file (10000000).

5.6.1 Data table format

The structure of the data file must be the following:

```
number of bodies to modify, 't', element symbol 1, elements symbol 2, ...
time 1, body 1 element 1, body 1 element 2, ...,
time 1, body 2 element 1, body 2 element 2, ...,
.
.
```

(continues on next page)

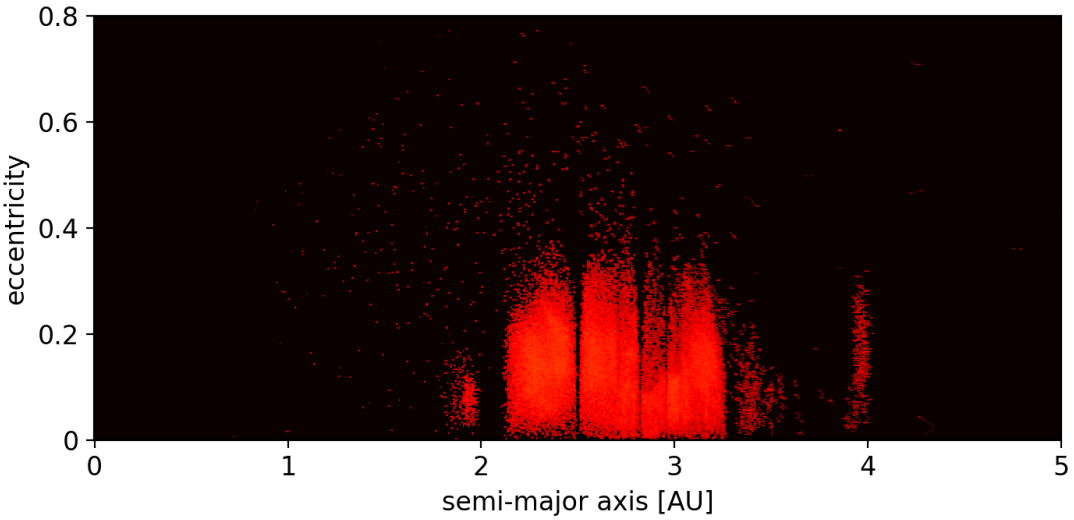

Fig. 5.5: a-e grid of the Solar System, including asteroids as test particles.

(continued from previous page)

```
.
time 2, body 1 element 1, body 1 element 2, ...,
time 2, body 2 element 1, body 2 element 2, ...,
.
.
.
```

with:

- The number of bodies 'n', indicates how many bodies will be modified. They are the first 'n' bodies in the initial condition file.

  The index of the planets can not be set. The order of the planets must correspond to the order of the initial conditions file.

- time is the time of the elements in years. The time must be included.
- elements can be:
  - a, semi-major axis in AU
  - e, eccentricity
  - i, inclination in radians
  - O, (Omega) longitude of the ascending node in radians
  - w, (omega) argument of periapsis in radians
  - M, Mean anomaly in radians
  - T, epoch time in days (Do not use together with M)
  - m, mass in Solar masses
  - r, radius in AU

- x, X-position in AU (heliocentric)
- y, Y-position in AU (heliocentric)
- z, Z-position in AU (heliocentric)
- vx, X-velocity in AU/day * 0.0172020989 (heliocentric)
- vy, Y-velocity in AU/day * 0.0172020989 (heliocentric)
- vz, Z-velocity in AU/day * 0.0172020989 (heliocentric)
- vxb, X-velocity in AU/day * 0.0172020989 (barycentric)
- vyb, Y-velocity in AU/day * 0.0172020989 (barycentric)
- vzb, Z-velocity in AU/day * 0.0172020989 (barycentric)
- -, skip that column

An example data file to modify the mass and radius of a single body looks like this:

```
1 t m r
0.0000000000000000 3.0024584e-7  2.7582675517426333e-5
2.2000000476840000 3.00262253e-7 2.75828934594789e-5
5.3680003681180004 3.00285888e-7 2.7583469524195 57e-5
9.9299211920929995 3.00319926e-7 2.7584299066671142e-5
.
.
.
```

An example data file to modify the semi-major axis, eccentricity and inclination of the first four bodies looks like the following, where the columns are: time, semi major axis, eccentricity and inclination:

```
4 t a e i
0    5.49973  3.17077e-05  1.03555e-06
0    5.70011  3.10758e-05  0.00546965
0    9.9999   3.09719e-06  0.000956204
0    11.25    2.33299e-06  0.00194193
100  5.49963  3.09278e-05  1.00262e-06
100  5.70002  3.01496e-05  0.00527889
100  9.99984  6.81447e-06  0.000935937
100  11.2499  7.15947e-06  0.00190979
200  5.49954  9.73926e-05  4.8273e-06
200  5.69991  9.89094e-05  0.00507701
200  9.99975  7.85054e-06  0.000913772
200  11.2498  7.8151e-06   0.00187511
300  5.49943  9.31021e-05  4.63857e-06
300  5.69978  9.42067e-05  0.00486796
300  9.99967  1.0987e-05   0.000890745
300  11.2497  1.12345e-05  0.00183792
.
.
.
```

5.7 Use single precision in kick forces

By default, GENGA uses double precision variables to integrate all particles. But GENGA offers the option to use single precision variables in the calculation of the gravitational force between the particles. This is particular interesting because non Tesla GPUs are much faster in calculating single precision than double precision operations. For large N simulations, the calculation of the N^2 gravitational force terms are often the dominant part. Therefore reducing the precision in these terms can give a significant speed up in the simulations. However, one must be sure that the reduced precision is still good enough to perform the given simulation.

In detail it is the acceleration of particle j on particle i, that can be calculated either in double or in single precision:

$$\mathbf{a}_{ij} = \frac{Gm_j}{r_{ij}^3}\mathbf{r}_{ij} \times K(r_{ij}),$$

Where $K(r_{ij})$ is the changeover function. Note that during a close encounter, the forces in the Bulirsch-Stoer integration are always computed in double precision.

In Fig. 5.6 is shown a speed comparison between single precision and double precision for different GPU types.

The use single precision in the gravitational forces instead of double precision, the `Do Kick in single precision` option in the *param.dat* file must be set to 1.

This option is not yet supported in the multi simulation mode.

5.8 Exact reproducible results

Usually, the outcome of a parallel numerical calculation is not exactly reproducible. The reason is, that calculations are done with a finite precision and that the outcome depends on the order of the operations. In general, we have $a + b + c \neq a + c + b$. And since the planetary N-body problem is chaotic, already tiny differences in the calculation can lead to a different result on individual bodies. Even the number of formed planets can vary from simulation to simulation. A detailed study about this effect is given in [HoffmannGrimmMooreStadel17].

However it is possible to force GENGA to exactly reproduce a given outcome. This is possible because most parallel operations are using parallel reduction sums with a fixed order. The only place which has not a fixed order, is the creation of the close encounter pair lists. By introducing an additional sorting step on the close encounter pairs lists, also this order can be fixed and the result of a given initial condition is always the same. It is important to note that this does not mean that the results are 'true', they still suffer from small round-off errors, just this error is always the same. Since the additional sorting step introduces also a performance penalty, it is not recommended to use this mode of GENGA for production runs. But it can be very useful to check if a GPU works correctly and it allows also to eliminate memory leaks in a code.

The enable the exact reproducible outcome mode on GENGA, the `Serial Grouping` option in the *param.dat* file must be set to 1.

5.9 Poincare surface of section

By setting the `def_poincareFlag` *define.h* file to 1, GENGA prints the Poincare surface of section. It prints the coordinates of x and v when a particle crosses the positive x coordinate. Note that this works only using the second order integrator, and not for test particles or in the multi simulation mode. An example of the surface of section of 32 planetesimals can be found [here](https://www.youtube.com/watch?v=a_4cjXVDEAw). Also note that by enabling this option, the integration speed is reduced.

The coordinates of the Poincare surface of section are reported in the *The Poincare surface of section file: Poincare<name><timeInterval>.dat*.

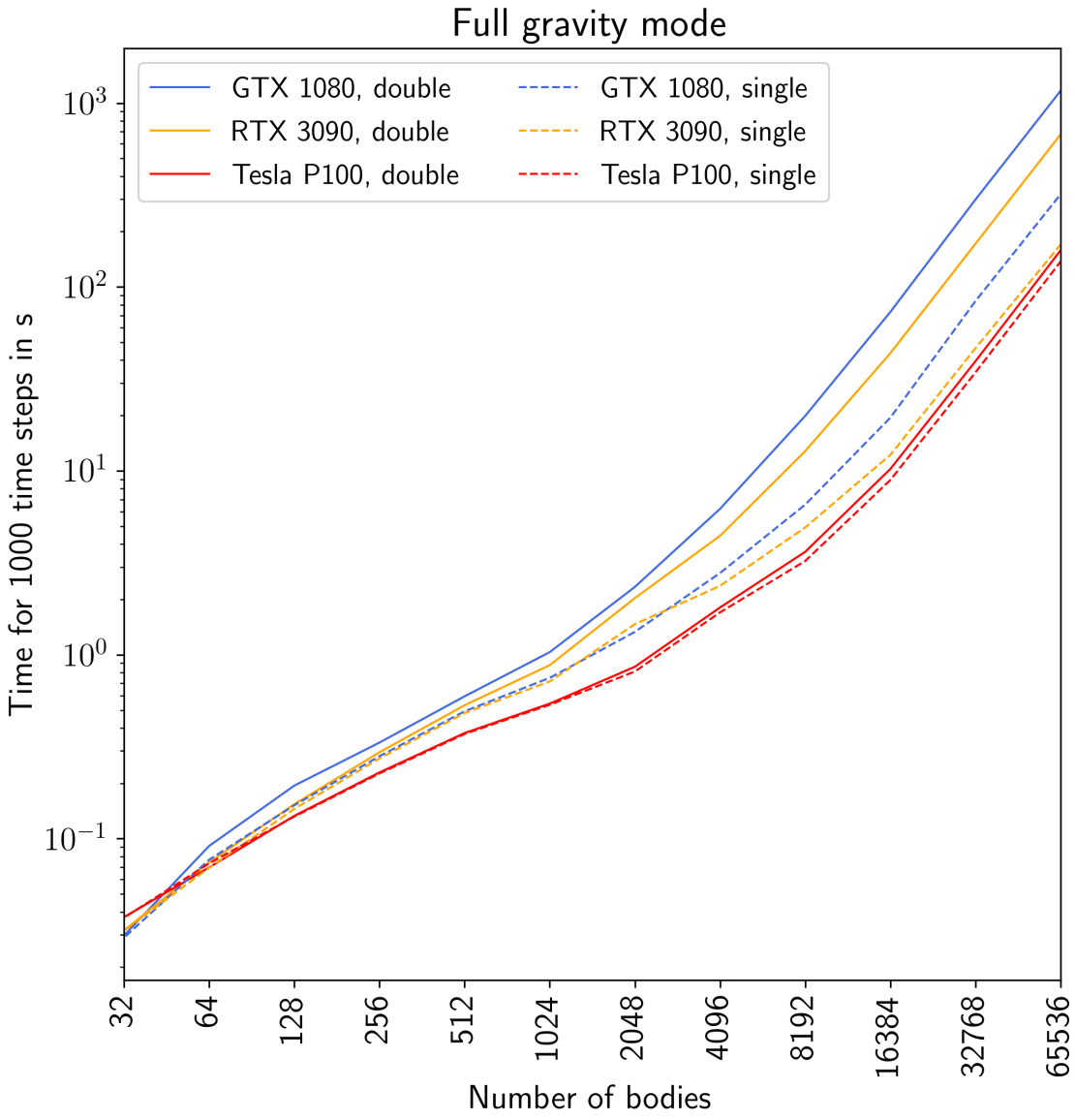

Fig. 5.6: Performance comparison between double precision and single precision force calculations. Especially non-Tesla cards can run much faster in single precision.

CHAPTER

SIX

FORCES

6.1 Gas disk

The gas disk is implemented according to [MorishimaStadelMoore10]. It supports gas drag, Type I migration and drag enhancement for small particles. The parameters for the gas disk can be set with the following arguments:

In the *param.dat* file

- `Use gas disk`
- `Use gas disk potential`
- `Use gas disk enhancement`
- `Use gas disk drag`
- `Use gas disk tidal dampening`
- `Gas dTau_diss`
- `Gas disk inner edge`:
- `Gas disk outer edge`:
- `Gas disk grid outer edge`:
- `Gas disk grid dr`:
- `Gas Sigma_10`
- `Gas alpha`
- `Gas beta`
- `Gas Mgiant`
- `Gas file name`

In the *define.h* file

- `def_Gasnz_g`: Number of cells in z direction for gas grid
- `def_Gasnz_p`: Number of cells in z direction for particle grid
- `def_h_1`: scale height at 1AU for c = 1km/s
- `def_M_Enhance`: factor for enhancement
- `def_Mass_pl`: factor for enhancement
- `def_fMass_min`: factor for enhancement
- `def_Gas_cd`: numerical gas drag coefficient

6.1.1 Gas disk structure

The gas disk structure is implemented as a uniform disk in space, which decays exponentially in time ([MorishimaStadelMoore10]).

$$\Sigma_{gas}(r,t) = \Sigma gas, 0 \left(\frac{r}{1AU}\right)^{-\alpha} \exp\left(-\frac{t}{\tau_{decay}}\right), \tag{6.1}$$

with the gas surface density at 1 AU $\Sigma gas, 0$, the dissipation time τ_{decay} and the power law exponent α.

The scale height of the gas disk is set to

$$h(r) = h_0 \frac{r}{1AU} \left(\frac{r}{1AU}\right)^{\beta}, \tag{6.2}$$

with the scale height at 1AU h_0.

Gas disk physical range in r

The range of the gas disk can be set with the `Gas disk inner edge:` and `Gas disk outer edge:` parameters. Outside of these boundaries, the gas disk density is 0.

Gas disk grid

In order to calculate the gas disk gravitational effect on the particles, the gas disk gravitational force in r and z is tabulated and stored in a gas disk grid. While the tabulated values of the grid respect the entire gas disk, ranging from the inner edge to the outer edge, the gas disk itself can have a smaller range in r. This is especially useful, when the gas disk extends a broader range than the particles. Therefore, the outer range of the gas disk grid can be set by a different parameter `Gas disk grid outer edge:`. The inner edge of the gas disk grid corresponds to the physical inner edge of the disk, `Gas disk inner edge:`

The spacing of the gas disk grid in r can be set with the parameter `Gas disk grid dr`.

When a particle is located outside of the gas disk grid, then the effect of the gas disk potential on the particles is not applied by using the tabulated values, but with a simpler approach according to Ward(1981).

6.1.2 Gas drag force

The gas drag force is enabled with the `Use gas disk drag` parameter. The gas drag force is implemented as

$$\mathbf{F}_{drag} = -\frac{1}{2m} c_D \pi r^2 \rho_{gas} |\mathbf{v}_{rel}| \mathbf{v}_{rel}, \tag{6.3}$$

with the radius of the particle r and the mass of the particle m. The numerical coefficient c_D is set to 2 ([MorishimaStadelMoore10]). The value of c_D can be changed in the *define.h* file.

6.1.3 Gas disk file

When a gas disk file name is specified in the `Gas file name` parameter. Then the gas disk structure is red from this file. The file must contain the following columns:

```
time0 r Sigma h
time1 r Sigma h
.
.
.
```

with:

- time in years.

- r the distance from the cell to the star in AU.
- Sigma, the surface density at the cell location r,in in g/cm^3.
- h, the gas disk scale height at the cell location r, in AU.

6.2 General Relativity Corrections

General Relativity corrections can be enabled with the `Use GR` parameter in the *param.dat* file. Three different methods of GR corrections are available:

- Hamiltonian Splitting
- Implicit midpoint
- Direct force

6.2.1 Hamiltonian splitting (UseGR = 1)

The Hamiltonian splitting method is implemented according to [SahaTremaine94]. The GR (or also called post Newtonian) Hamiltonian takes the form:

$$H_{PN} = \sum_i \left(\alpha_i H_{Kep,i}^2 + \frac{\beta_i}{r_i^2} + \gamma_i p_i^4 \right) \tag{6.4}$$

with

$$\alpha_i = \frac{3}{2 m_i c^2},$$

$$\beta_i = \frac{-\mu_i^2 m_i}{c^2},$$

$$\gamma_i = -\frac{1}{2 m_i^3 c^2},$$

$$\mu_i = G(M_\star + m_i),$$

and the speed of light c.

Following [SahaTremaine94], the γ term is expressed as

$$d\mathbf{x}_i = -dt \cdot 2 \frac{v_i^2}{c^2} \mathbf{v}_i,$$

with the time step dt. This term is combined with the Sun-kick term.

The β term can be expressed as

$$d\mathbf{v}_i = -dt \cdot 2 \frac{\mu^2}{c^2 r_i^4} \mathbf{r}_i,$$

and is combined with the Kick operation, but not affected by the changeover function.

The α term is implemented through a time step modification in the particle drift operation (and also the close encounter direct integrator):

$$dt_i' = dt \cdot \left(1 - \frac{3}{2} \frac{\mu}{a_i c^2} \right).$$

To apply the described GR Hamiltonian splitting, the velocities must be converted into pseudovelocities. Since GENGA allows the use of other non-Newtonian forces, which also can depend on the velocities, we need to compute those terms in real velocities and not pseudovelocities. Therefore we convert at each time step from real velocities to pseudovelocites and back. Only the Sun-kick and the drift operation is done in pseudovelocities.

6.2.2 Implicit Midpoint (UseGR = 2)

The acceleration of the GR (post-Newtonian) effect can be approximated following [Kidder95] [MardlingLin02] or [Fabrycky10] as:

$$\begin{aligned}\mathbf{a}_{PN} = -\frac{G(M_\star + m_i)}{r_i^2 c^2} \cdot \Bigg\{ &- 2(2-\eta)\dot{r}_i \dot{\mathbf{r}}_i \\ &+ \left[(1+3\eta)\dot{\mathbf{r}}_i \cdot \dot{\mathbf{r}}_i - \frac{3}{2}\eta \dot{r}_i^2 - 2(2+\eta)\frac{G(M_\star + m_i)}{r_i}\right] \hat{\mathbf{r}} \Bigg\},\end{aligned} \tag{6.5}$$

where c is the speed of light and η is defined as:

$$\eta = \frac{M_\star m_i}{(M_\star + m_i)^2}.$$

Since the equation (6.5) depends on the positions and the velocities, the GR corrections must by applied with the implicit midpoint method. In that way the symplectic nature of the integration is conserved.

6.2.3 Direct Force (UseGR = 3)

It is also possible to apply the equation (6.5) without the implicit midpoint method. But then the integration is not fully symplectic and the semi-major of the affected bodies is not constant. Therefore this mode is mainly for testing and for comparisons.

6.2.4 Test and comparison

A good object to test the general relativity effect is the asteroid (1566) Icarus. It has an orbital eccentricity of 0.83 which leads to a closed approach to the Sun of about 0.2 AU. In order to test the orbit or Icarus, we integrate it together with the 16 most massive objects of the Solar System. All data are taken from the JPL HORIZON system. We compare our integration with the measured orbit of Icarus. The results are shown in Fig. 6.1. One can clearly see how the GR corrections improve the orbital positions during the first few orbital periods. Comparing the Hamiltonian splitting approach with the implicit midpoint method leads to no significant difference. The integration precision can be improved, when the 16 perturber objects are not integrated, but rather interpolated from the known ephemeris. The remaining difference in the orbital position after applying the GR corrections is most likely caused by other non-gravitational effects. Or also small deviations in the initial conditions get increased at each orbit.

Since the GR effect calculation scales only linearly with the number of particles N, the performance of large simulations ($N \gtrapprox 4096$) is only affected marginally. Simulations with $N \approx 1024$ are affected by ca. 1%, while small simulations are affected more. A simulation with 16 fully interacting bodies can take up to 1.5 times as long with implicit midpoint method, and more than twice as long with the Hamiltonian splitting method. We recommend using the implicit midpoint method, since it gives a better performance. In Figure Fig. 6.2 is shown a comparison of the performance between the GR modes.

6.3 Tidal Forces

Tidal Forces can be enabled with the `Use Tides` parameter in the *param.dat* file. The tidal forces evolve the spin of the particles and spin of the central star. The evolution of the stellar spin is reported in *The stellar evolution file: Star<name>.dat*.

Parameters for the tidal forces:

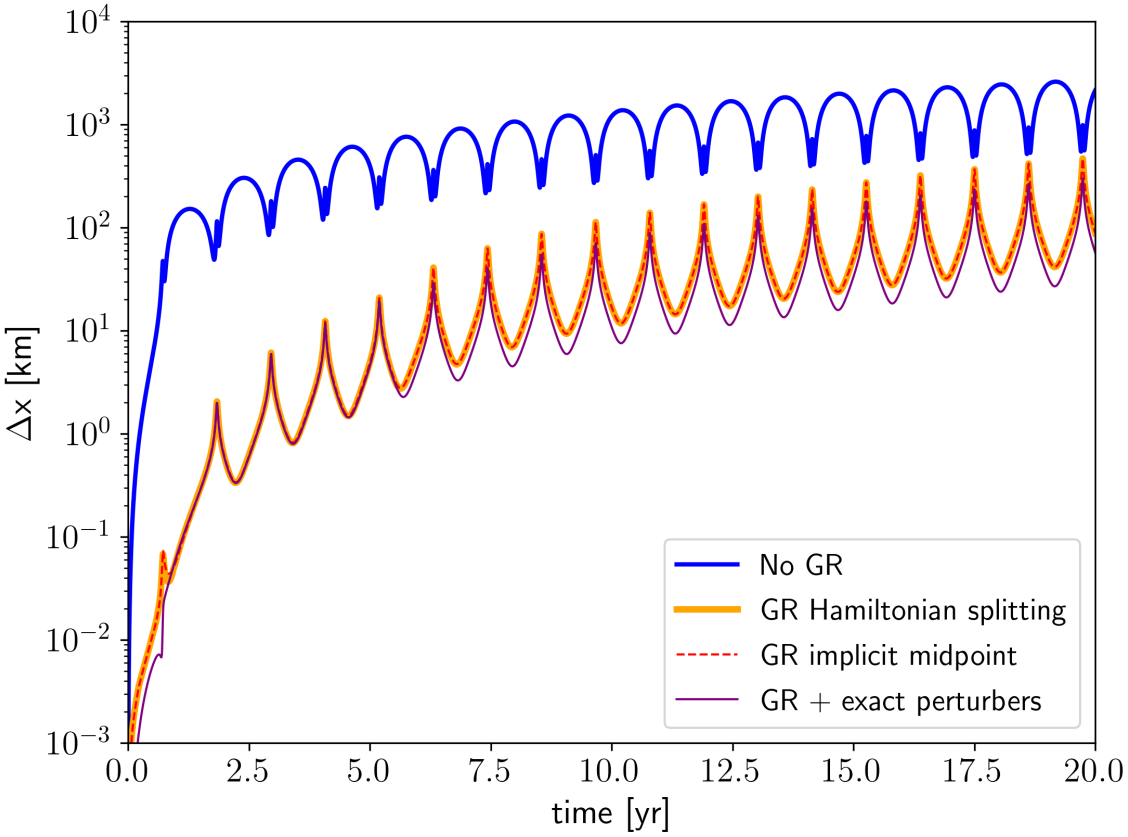

Fig. 6.1: Difference in position of the asteroid (1566) Icarus between the integration and the measured orbit. The Asteroid Icarus is integrated together with the 16 most massive objects of the Solar System. When GR corrections are not considered (blue line) then the position after two orbital periods differs by more than 100 km. With GR effects enabled, the difference is less than two km. There is not significant difference between the Hamiltonian splitting approach (orange line) and the implicit midpoint method (red dashed line). When the position of the 16 perturbers is not integrated, but interpolated from their measured positions, then the difference in position is reduced further (purple line).

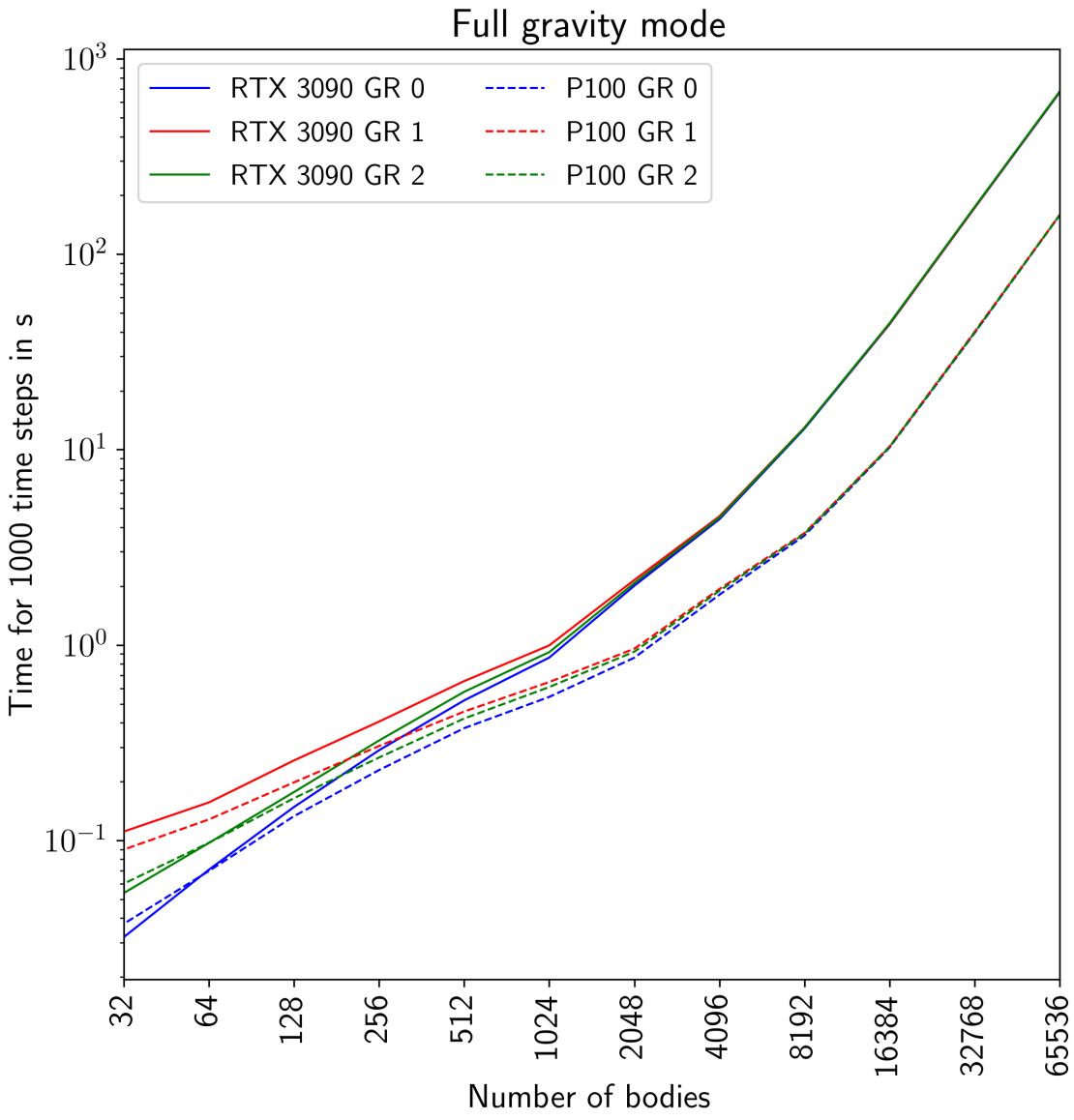

Fig. 6.2: Performance of the GR modes on two GPUs for a set of N fully interactive particles.

- `Star Love Number`
- `Star tau`
- `Star spin_x`
- `Star spin_y`
- `Star spin_z`
- `Star Ic`

and the initial conditions:

- k2
- tau
- Sx
- Sy
- Sz
- Ic

For implementing the tidal effects, we follow the weak friction tidal model described in [Hut81] and [BolmontRaymondLeconte+15]. In this model, the tidal bulge of a primary body with radius R, induced by a companion body of mass m, is described by two point masses at the surface of the primary body with mass [Hut81]

$$\mu \approx \frac{1}{2} k_{2,i} m R^3 \left[r(t-\tau)\right]^{-3},$$

where $r(t-\tau)$ is the distance between the two bodies at time $t-\tau$, and τ is a constant time lag, which models the tidal dissipation. $k_{2,i}$ is the potential Love number of degree 2 of the body i.

The tidal force can be written as:

$$\begin{aligned}\mathbf{F}_T = \left[-(D_{t0,\star} + D_{t0,i}) - 2\frac{\dot{r}_i}{r_i}(P_{t0,\star} + P_{t0,i})\right] \mathbf{e}_{ri} \\ + P_{t0,i}\left(\mathbf{\Omega}_i \times \mathbf{e}_{ri}\right) + P_{t0,\star}\left(\mathbf{\Omega}_\star \times \mathbf{e}_{ri}\right) \\ - \left(P_{t0,i} + P_{t0,\star}\right)\frac{\mathbf{v}_i}{r_i}.\end{aligned}$$

with the quantities:

$$D_{t0,i} = 3G\frac{m_\star^2 R_i^5}{r_i^7} k_{2,i}$$

$$D_{t0,\star} = 3G\frac{m_i^2 R_\star^5}{r_i^7} k_{2,\star}$$

and

$$P_{t0,i} = D_{t0,i}\tau_i$$

$$P_{t0,\star} = D_{t0,\star}\tau_\star.$$

where $\mathbf{e}_{ri}$ is the radial unit vector, and $\mathbf{\Omega}_\star$ is the rotational angular velocity vector of the star, $\mathbf{\Omega}_i$ the rotational angular velocity vector of the body i.

(See *Spin* to convert the rotational angular velocity into a spin.)

Since the equation (6.5) depends on the positions and the velocities, the GR corrections must by applied with the implicit midpoint method. In that way the symplectic nature of the integration is conserved.

6.3.1 Tidal spin evolution

Additionally to the acceleration on the particles, the tidal force generates a torque, which changes the spin of the particles and of the central star.

This tidal torque is given as

$$\mathbf{N}_T = \mathbf{r} \times \mathbf{F}_{T\theta},$$

with the transverse component of the tidal force $\mathbf{F}_{T\theta}$.

With the relation

$$\mathbf{r} \times (\mathbf{\Omega} \times \mathbf{e}_r) - \mathbf{r} \times \frac{\mathbf{v}}{r} = \mathbf{\Omega} r - (\mathbf{r} \cdot \mathbf{\Omega})\mathbf{e}_r - \mathbf{e}_r \times \mathbf{v},$$

the tidal torque can be written as:

$$\mathbf{N}_{Ti} = P_{to,i} \left[\mathbf{\Omega}_i r_i - (\mathbf{r}_i \cdot \mathbf{\Omega}_i)\mathbf{e}_{ri} - \mathbf{e}_{ri} \times \mathbf{v}_i\right]$$

$$\mathbf{N}_{T\star} = P_{to,\star} \left[\mathbf{\Omega}_\star ri - (\mathbf{r}_i \cdot \mathbf{\Omega}_\star)\mathbf{e}_{ri} - \mathbf{e}_{ri} \times \mathbf{v}_i\right]$$

By using heliocentric coordinates, the spin evolution is takes the form [BolmontRaymondLeconte+15]

$$\frac{d}{dt}(I_\star \mathbf{\Omega}_\star) = -\sum_{j=1}^{N} \frac{m_\star}{m_\star + m_j} \mathbf{N}_{T\star}$$

and

$$\frac{d}{dt}(I_i \mathbf{\Omega}_i) = -\frac{m_\star}{m_\star + m_i} \mathbf{N}_{Ti},$$

with the moment of inertia I.

6.3.2 Test and comparison

To test the implementation of the tidal forces, we repeat the calculations of the Mars-Phobos system, described, in [Efroimsky07] and [AuclairDesrotourLePLafitteMathis14]. As initial values, we use the same parameters as given in Table 1 in [AuclairDesrotourLePLafitteMathis14]. Additionally, we use an eccentricity of 0.0151.

6.4 Rotational Deformation

The rotational deformation force can be enabled with the `Use Rotational Deformation` parameter in the *param.dat* file.

Parameters for the tidal forces:

- `Star fluid Love Number`
- `Star spin_x`
- `Star spin_y`
- `Star spin_z`
- `Star Ic`

and the initial conditions:

- k2f
- Sx

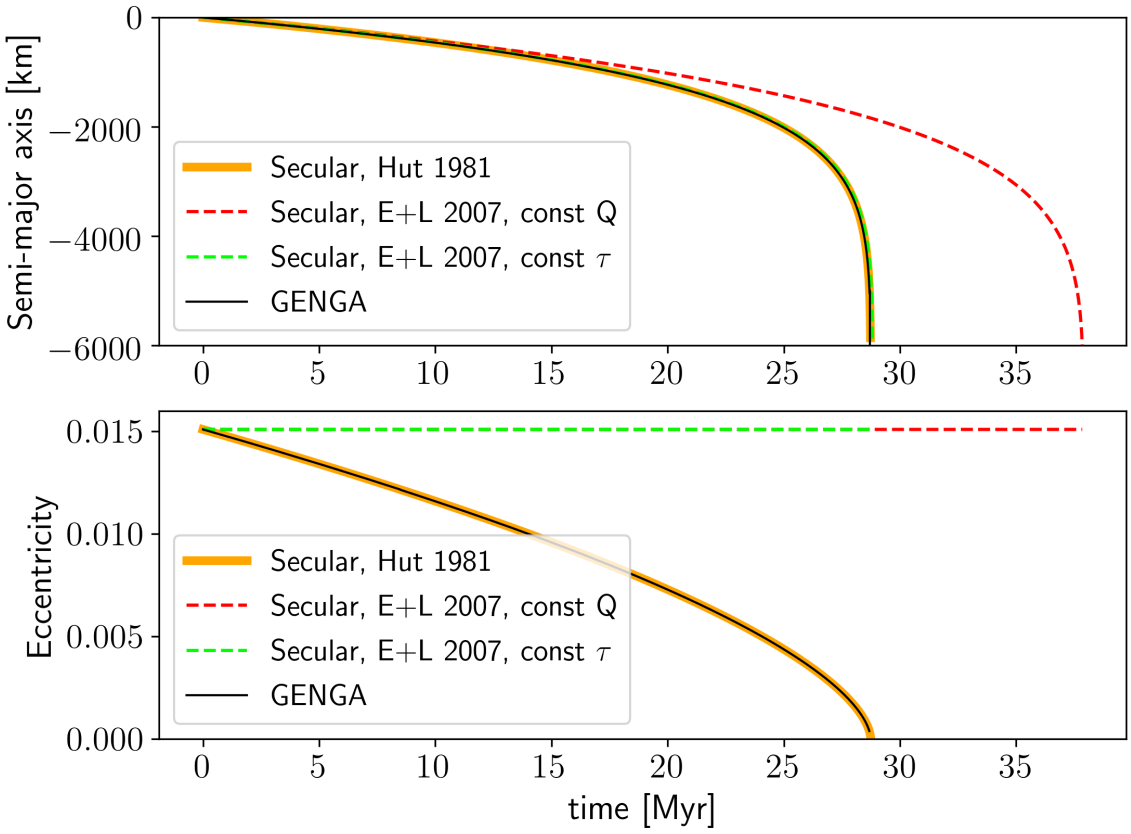

Fig. 6.3: Comparison of GENGA against secular evolution models on the example of the Mars-Phobos system. The models correspond to [Hut81] (blue solid line), (Equation 30, [Efroimsky07]) (red dashed line) and (Equation 36, [Efroimsky07])(green dashed line). The evolution of the semi-major axis and the eccentricity in GENGA agree well with the constant τ scenario in the secular evolution models. This plot reproduces the results of Figure 2 from [Efroimsky07].

- Sy
- Sz
- Ic

When a viscous body is rotating, then its shape is transformed into a symmetric oblate ellipsoid. The potential energy of a system containing two oblate bodies 0 and 1 is give in [Moy71], Equation 158, [Moy03] or [CorreiaLaskarFaragoBoue11] as

$$U = -\frac{Gm_0m_1}{r}\left[1 - \sum_{i=0,1}\sum_{n=1}^{\infty} J_n \left(\frac{R_i}{r}\right)^n Pn(\hat{r}\cdot\hat{\Omega}_i)\right],$$

with the mass m, the physical radius R, the distance between the bodies r, the rotational angular velocity Ω and the Legendre polynomial of degree n, $Pn(x)$.

Further, we truncate the order n to two and introduce the J_2 parameter [CorreiaLaskarFaragoBoue11], [BolmontRaymondLeconte+15]

$$J_{2,i} = k_{2f,i}\frac{\Omega_i^2 R_i^3}{3Gm_i}$$

and

$$J_{2,\star} = k_{2f,\star}\frac{\Omega_\star^2 R_\star^3}{3Gm_\star},$$

with $k_{2f,i}$ the second potential Love number (fluid Love number).

We follow the description in [BolmontRaymondLeconte+15] and define the following quantities (In [BolmontRaymondLeconte+15], $C_\star$ and C_i are reversed):

$$C_\star = \frac{1}{2}Gm_im_\star J_{2,\star}R_\star^2$$

and

$$C_i = \frac{1}{2}Gm_im_\star J_{2,i}R_i^2.$$

Then the force due to the rotational deformation is given as [BolmontRaymondLeconte+15]:

$$\begin{aligned}\mathbf{F_R} = \Bigg\{&-\frac{3}{r_i^5}(C_\star + C_i)\\ &+\frac{15}{r_i^7}\left[C_\star\frac{(\mathbf{r}_i\cdot\mathbf{\Omega}_\star)^2}{\mathbf{\Omega}_\star^2} + C_i\frac{(\mathbf{r}_i\cdot\mathbf{\Omega}_i)^2}{\mathbf{\Omega}_i^2}\right]\Bigg\}\mathbf{r}_i\\ &-\frac{6}{r_i^5}\left(C_\star\frac{\mathbf{r}_i\cdot\mathbf{\Omega}_\star}{\mathbf{\Omega}_\star^2}\mathbf{\Omega}_\star + C_i\frac{\mathbf{r}_i\cdot\mathbf{\Omega}_i}{\mathbf{\Omega}_i^2}\mathbf{\Omega}_i\right).\end{aligned}$$

6.4.1 Rotational deformation spin evolution

Additionally to the acceleration on the particles, the rotational deformation force generates a torque, which changes the spin of the particles and of the central star.

This rotational deformation torque is given as

$$\mathbf{N}_R = \mathbf{r}\times\mathbf{F}_{R\theta},$$

with the transverse component of the tidal force $\mathbf{F}_{\mathbf{r}\theta}$.

It can be written following [BolmontRaymondLeconte+15] as

$$\mathbf{N}_{R\star} = -\frac{6}{r_i^5} C_\star \frac{\mathbf{r}_i \cdot \mathbf{\Omega}_\star}{\mathbf{\Omega}_\star^2} \left(\mathbf{r}_i \times \mathbf{\Omega}_\star\right)$$

and

$$\mathbf{N}_{Ri} = -\frac{6}{r_i^5} C_i \frac{\mathbf{r}_i \cdot \mathbf{\Omega}_i}{\mathbf{\Omega}_i^2} \left(\mathbf{r}_i \times \mathbf{\Omega}_i\right).$$

By using heliocentric coordinates, the spin evolution is takes the form [BolmontRaymondLeconte+15]

$$\frac{d}{dt}(I_\star \mathbf{\Omega}_\star) = -\sum_{j=1}^{N} \frac{m_\star}{m_\star + m_j} \mathbf{N}_{R\star}$$

and

$$\frac{d}{dt}(I_i \mathbf{\Omega}_i) = -\frac{m_\star}{m_\star + m_i} \mathbf{N}_{Ri},$$

with the moment of inertia I.

6.5 J2: additional gravitation multipole expansion

An additional gravitational multipole expansion term can be added to the simulation with the J2 force.

The J2 force can be enabled by setting the `J2` parameter in the *param.dat* file to any value different than 0.

Parameters for the J2 forces:

- `J2`
- `J2 radius` (mean radius of mass distribution, R_{mean})

The potential energy of an additional gravitational multipole expansion is given in [ZdericMadigan20], Equation 1, as

$$U_i = -\frac{G m_\star m_i}{r} \left[1 - J_2 \left(\frac{R_{\text{mean}}}{r}\right)^2 P_2(\hat{r} \cdot \hat{\Omega}_{\text{system}})\right],$$

with the mass of the central star $m_\star$, the mass of a particle i m_i, the mean radius of the mass distribution R_{mean}, the distance between the body and the star r, the rotational angular velocity of the N-body system Ω_{system} and the Legendre polynomial of degree 2, $P_2(x) = \frac{1}{2}(3x^2 - 1)$.

The value of Ω_{system} is assumed to be (0,0,1), corresponding to be spin aligned to the N-body system.

The J_2 value corresponds to [ZdericMadigan20]

$$J_2 = \frac{1}{2.0 m_\star R_{\text{mean}}^2} \sum_{i=0}^{n} \left(m_i a_i^2\right)$$

The implementation of the force is done similarly as the *Rotational Deformation* force.

6.6 Yarkovsky effect

The Yarkovky effect is implemented according to [VokrouhlickyFarinella99] and [VokrouhlickyMilaniChesley00] and is available in the schemes (we recommend scheme 1):

- Velocity kick $\mathbf{a_Y}$
  set `Use Yarkovsky = 1` in the *param.dat* file.

See Equation (6.6) and (6.7)

- Time averaged change in sami-major axis $\frac{da}{dt}$
  set `Use Yarkovsky = 2` in the *param.dat* file.
  See Equation (6.8) and (6.9)

The following parameters are relevant for the Yarkovsky effect and can be set in the *param.dat* file:

- `Use Yarkovsky`
- `Solar Constant`: Solar Constant at 1 AU in W / m^2
- `Asteroid eps`: Emissivity factor
- `Asteroid rho`: density of the body in kg/ m^3
- `Asteroid C`: Specific Heat Capacity in $J\,kg^{-1}K^{-1}$
- `Asteroid A`: Bond albedo
- `Asteroid K`: Thermal conductivity in $W\,m^{-1}K^{-1}$

Note that the calculation of the Yarkovsky effect uses the `Asteroid rho` value for the calculation of the thermal inertia Γ and not the individual particle densities.

$$\mathbf{a}_{seasonal} = \frac{4}{9}\frac{(1-A)\Phi}{(1+\lambda)} \times \sum_{k\geq 1} G_k \left[s_P \alpha_k \cos(knt + \delta_k) + s_Q \beta_k \sin(knt + \delta_k) \right] \mathbf{s}, \tag{6.6}$$

$$\mathbf{a}_{diurnal} = \frac{4}{9}\frac{(1-A)\Phi}{(1+\lambda)} G \left[\sin\delta + \cos\delta \mathbf{s}\times \right] \frac{\mathbf{r}\times\mathbf{s}}{r}. \tag{6.7}$$

$$\left(\frac{da}{dt}\right)_{diurnal} = -\frac{8}{9}\frac{(1-A)\Phi}{n}\frac{G\sin\delta}{1+\lambda}\cos\gamma \tag{6.8}$$

$$\left(\frac{da}{dt}\right)_{seasonal} = \frac{4}{9}\frac{(1-A)\Phi}{n}\sum_{k\geq 1}\frac{G_k \sin\delta_k}{k}\chi_k\bar{\chi_k}, \tag{6.9}$$

6.6.1 Set the call interval

With the parameter `Yarkovky Inteval` the calling interval of the Yarkovsky function can be set. This means that the function is called fewer times, and in increased drift rate $dx = a * dt * YarkovskyInterval$. With the time averaged Yarkovsky function (mode 2) it is possible to use quite large interval numbers. The direct Yarkovsky effect (mode 1) is more sensitive to values greater than ≈ 100. Strictly speaking ,this argument violets the symplectic nature of the integrator, but as long as the Yarkovky force is small it is OK.

6.6.2 Test of the Yarkovsky effet

In Fig. 6.4 is shown a test of the Yarkovsky effect following [FarinellaVokrouhlickyHartmann98] and [BRB00].

Relevant parameters for this example:

- Use Yarkovsky = 1 (2)
- Asteroid emissivity eps = 1.0
- Asteroid density rho = 3500.0

- Asteroid specific heat capacity C = 680.0
- Asteroid albedo A = 0.0
- Asteroid thermal conductivity K = 2.65
- Solar Constant = 1367

Initial conditions:

- Semi-major axis a = 2 AU
- Eccentricity = 0
- Inclination = 0
- Argument of perihelion = 0 - 2 π
- Longitude of ascending node = 0 - 2 π
- Mean anomaly = 0 - 2 π
- Density ρ = 3500.0 kg/m^3
- Physical radius R = 0.1 - 100 m
- Rotation frequency ω = 5 h
- Obliquity $\gamma_{\text{Seasonal}} = 90^\circ$
- Obliquity $\gamma_{\text{Diurnal}} = 0$

6.7 Poynting-Robertson effect

The Poynting-Robertson effect consists of the Poynting-Robertson drag and the radiation pressure force. The Poynting-Robertson effect is implemented according to [BurnsLamySoter79] and is available in the schemes (we recommend scheme 1):

- Velocity kick $\mathbf{a_{PR}}$
  set `Use Poynting-Robertson = 1` in the *param.dat* file.
  See Equation (6.12) and (6.13)
- Orbital averaged change in semi-major axis and eccentricity $\frac{da}{dt}$, $\frac{de}{dt}$
  set `Use Poynting-Robertson = 2` in the *param.dat* file.
  See Equation (6.10)

The following parameters are relevant for the Poynting-Robertson effect and can be set in the *param.dat* file:

- `Use Poynting-Robertson`
- `Solar Constant`: Solar Constant at 1 AU in W /m^2
- `Radiation Pressure Coefficient Qpr` (in general assumed to be 1)

6.7.1 Solar Wind

Solar wind drag can be added to the Poynting-Robertson effect according to [LZJ95]. The solar wind drag is only applied in scheme 1. Enable it with:

- `Solar Wind factor`, only applied to scheme 1.

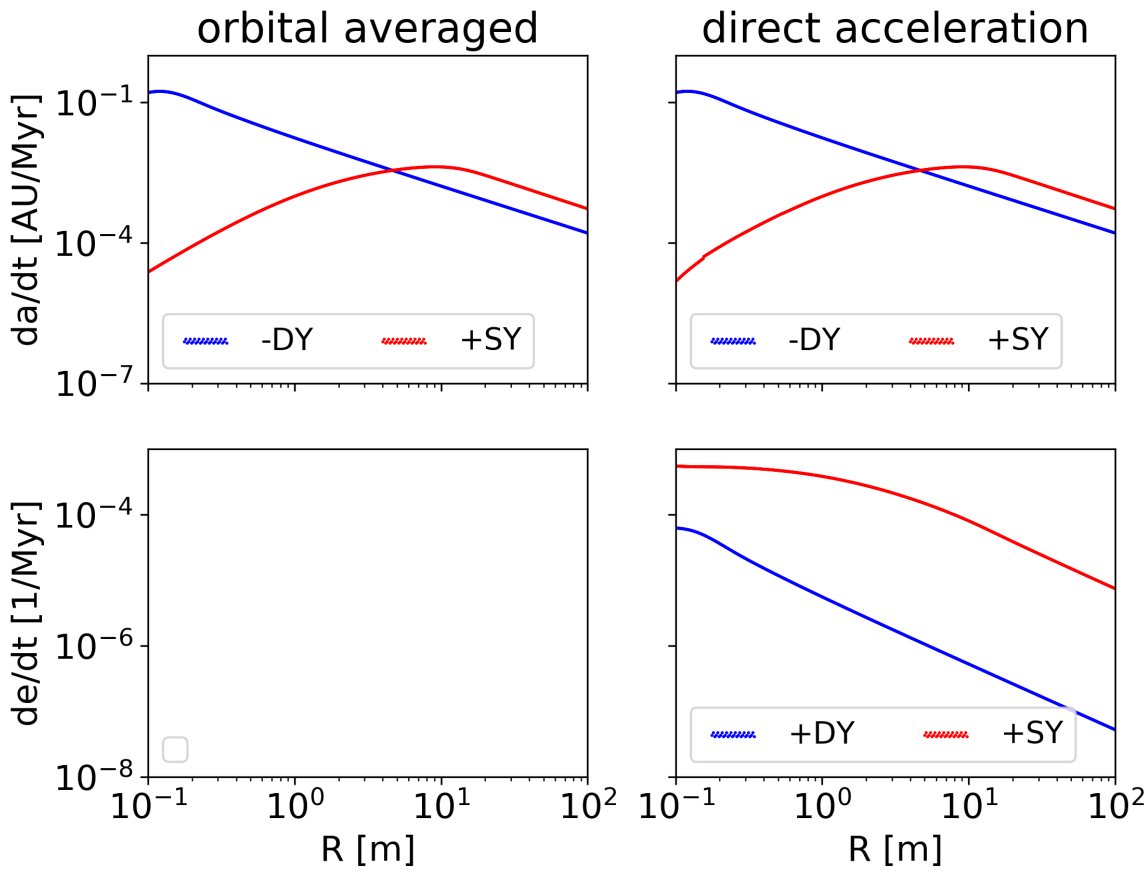

Fig. 6.4: Drift rate of the seasonal and diurnal Yarkovsky effect. Computed after 10000 years and averaged over 1 orbit.

6.7.2 Scheme 1: Velocity kick

The Poynting-Robertson effect is implemented according to the following equation:

$$\frac{d\mathbf{v}}{dt} = \frac{\eta c}{r^2} Q_{pr} \left[\left(1 - \frac{\dot{r}}{c} \right) \hat{r} - \frac{\mathbf{v}}{c} \right]. \tag{6.10}$$

When solar wind is included, then the following equation is used:

$$\frac{d\mathbf{v}}{dt} = \frac{\eta c}{r^2} Q_{pr} \left[\left(1 - (1+sw)\frac{\dot{r}}{c} \right) \hat{r} - (1+sw)\frac{\mathbf{v}}{c} \right]. \tag{6.11}$$

where sw is the ratio of solar wind drag to Poynting-Robertson drag.

6.7.3 Scheme 2: orbital averaged drift rates

$$\frac{da}{dt} = -\frac{\eta}{a} Q_{pr} \frac{(2+3e^2)}{(1-e^2)^{3/2}} \tag{6.12}$$

$$\frac{de}{dt} = -\frac{5}{2} \frac{\eta}{a^2} Q_{pr} \frac{e}{(1-e^2)^{1/2}}, \tag{6.13}$$

6.7.4 Set the call interval

With the parameter `Poynting-Robertson Inteval` the calling interval of the Poynting-Robertson function can be set. This means that the function is called fewer times, and in increased drift rate $dx = a * dt * Poynting - RobertsonInterval$. With the time averaged Poynting-Robertson function (mode 2) it is possible to use quite large interval numbers. The direct Yarkovsky effect (mode 1) is more sensitive to values greater than ≈ 100. Strictly speaking ,this argument violets the symplectic nature of the integrator, but as long as the Yarkovky force is small it is OK.

6.7.5 Test of the Poynting-Robertson effect

In Fig. 6.5 is shown a test of the Poynting-Robertson effect for the same initial conditions as in *Test of the Yarkovsky effet*, but with eccentricities $\neq 0$.

Relevant parameters for this example:

- Use Poynting-Robertson = 1 (2)
- Solar Constant = 1367
- Radiation Pressure Coefficient Qpr = 1

Initial conditions:

- Semi-major axis a = 2 AU
- Eccentricity = 0.05
- Inclination = 0
- Argument of perihelion = 0 - 2 π
- Longitude of ascending node = 0 - 2 π
- Mean anomaly = 0 - 2 π

- Density ρ = 3500.0 kg/m^3
- Physical radius R = 0.1 - 100 m

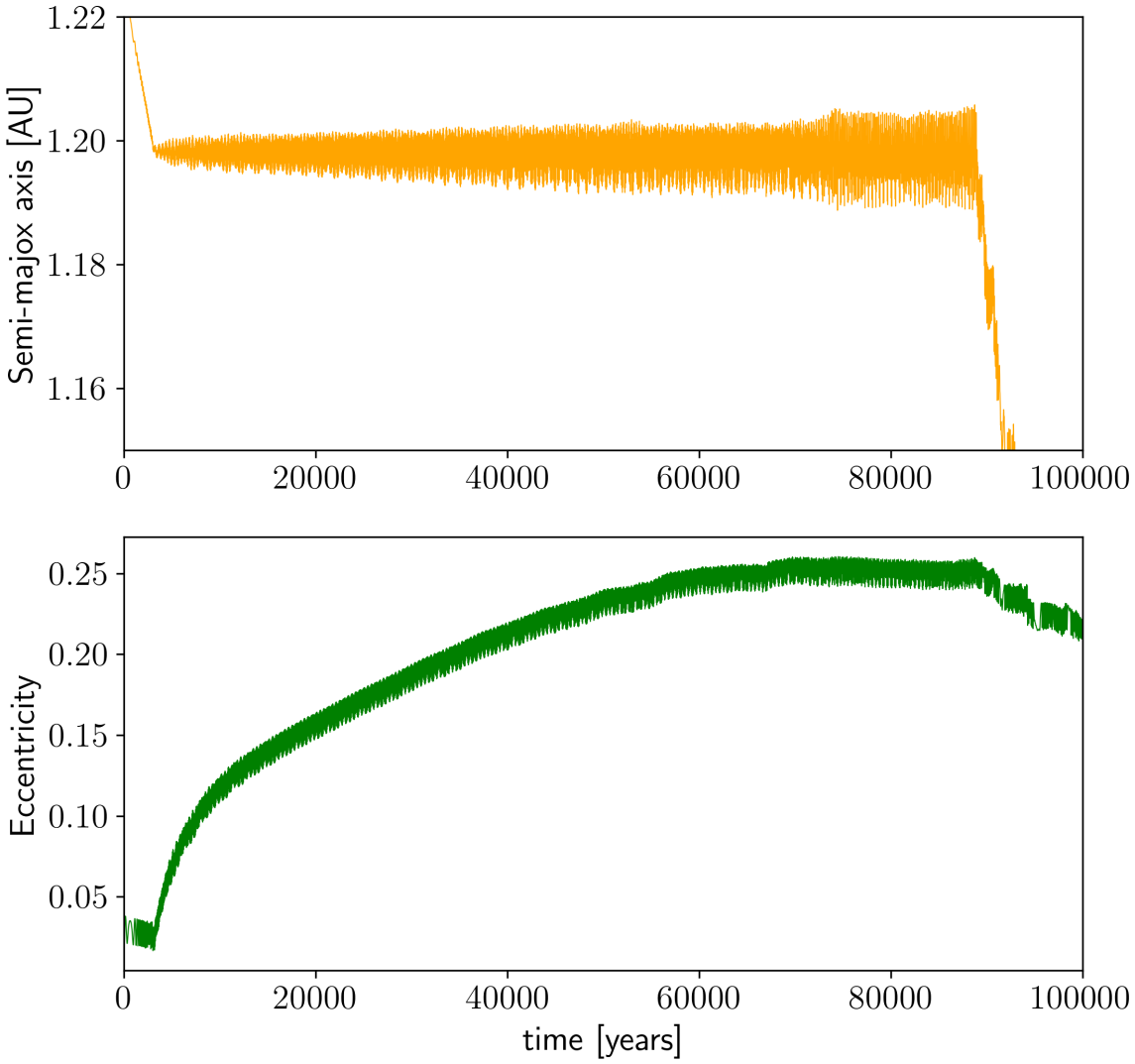

Fig. 6.5: Drift rate of the Poynting-Robertson effect. Computed after 10000 years and averaged over 1 orbit.

CHAPTER

SEVEN

SMALL BODY COLLISION MODEL

7.1 Model for small bodies collisions

Simulating small bodies of the Solar System (or other planetary systems) often requires the integration of a very large number of particles. There are more than 1 million asteroids known in the asteroid belt, and taking into account also the (unseen) smaller fragments would lead to immense numbers, exceeding current computational capabilities. But still the presence of all these small bodies is important for the dynamics of meteoroids: collisions can lead to both break-up events and changes in meteoroid rotation rate. Together with the Yarkovsky effect and Poynting-Robertson drag, collisions between small bodies can influence the migration rate of asteroids and thus generate impact events on the Earth or other planets. An alternative to simulate such collisions with an N-body integrator is to include a collision model, which uses an average probability that a given small body would collide with another small body [FarinellaVokrouhlickyHartmann98]. In the next two sections, we describe our model for collisional break-up events and rotation changes of small bodies.

The model for small bodies collisions can be enabled with the `Use Small Collisions` parameter in the *param.dat* file. It is only available for test particles.

All events of rotation reset or fragmentation are reported in the Fragments file *The Fragments File: Fragments<name>.dat*.

Since the fragmentation routine creates additional debris particles, the initial memory arrays need to be increased, to be able to store these additional particles.
That can be done with the `Nfragments`: option in the *param.dat* file.

The maximum number of particles in a simulation is the number of initial particles and test particles plus the value set in `Nfragments`.

$$N_{Maximum} = N(Initialbodies) + N(Initialtestparticles) + Nfragments$$

If the maximum number of allowed particles is reached, then no more new particles are created until some other particles are removed from the simulation by a collision or ejection event.

The following parameters are relevant for the small body collision model and can be set in the *param.dat* file:

- `Use Small Collisions`, enable small bodies collisions model.
- `Small Collisions Interval`, Interval at which the model is called.
- `Asteroid rho`: density of the body in kg/m^3.

- `Asteroid V`: Asteroid collisional velocity V, in m/s.
- `Nfragments`: Number of additional memory size for debris particles, in particle numbers, (default 0).
- `Asteroid minimal fragment radius`, in m, R_m value. (default = 0.01 m).
- `Asteroid fragment remove radius`, in m, R_{del} value. (default = 0.01 m).

7.1.1 Collisional break-up

Enabled with `Use Small Collisions = 1` or `Use Small Collisions = 3`.

Following [FarinellaVokrouhlickyHartmann98], we use a collisional life-time of

$$\tau_{col} = 2.0 \cdot 10^7 \sqrt{\frac{R}{1\text{m}}} \text{years},$$

which is equivalent to a break-up probability of $1/\tau_{col}$ per year. When a small body breaks up, its mass is replaced by a series of fragments with radii between the radius of the original object R_0, and R_m is set by `Asteroid minimal fragment radius`. We use the following distribution to generate the radii of the new fragments R_f:

$$R_f = \left[\left(R_0^{(n+1)} - R_m^{(n+1)} \right) \cdot y + R_m^{(n+1)} \right]^{1/(n+1)},$$

with n = -1.5 and a uniform generated random number $y \in (0, 1)$. The mass of each generated fragment is subtracted from the remaining mass of the original body. The first fragment to exceed the remaining mass gets the remaining mass.

For the velocity distribution of the fragments, we use a scaling value

$$u = y \cdot m_f^{-1/6},$$

with the mass of the fragment m_f, a random number $y \in (1 - a, 1 + a)$ and $a = 0.2$ [NF91].

A constant velocity budget of 31 m/s (based on $V_{budget} = (0.3/N)^{0.5} \times V_{impactor}$, and $N = 8000$; see [Wie15] for details) is distributed to the fragments in proportion to u. A randomly distributed velocity vector with the given length sets the new orbit of the particle. Typical values of these created fragment velocities are of the order of $\sim 0.1 - 1$ m/s.

Fragments with a radius smaller than R_{def} are removed to prevent having too many particles in the simulation. The obliquity of the new particles is generated randomly (between 0 and π) and for the rotation rate, we take

$$\omega_f = random \left(\frac{1}{36 R_f}, \frac{1}{R_f} \right),$$

which reproduces the range of rotation rates typically observed for Near Earth Asteroids in the meter [BB00] to kilometer [FarinellaVokrouhlickyHartmann98] size range.

7.1.2 Collisional reset of rotation rate and obliquity

Enabled with `Use Small Collisions = 1` or `Use Small Collisions = 2`.

For each object at each time step, we calculate the probability that its rotation was reset during the last time step by a virtual collision with another particle, as a function of its radius R and previous rotation rate. The rotation will be changed if the impactor has an angular momentum at the collision time similar to the rotational angular momentum of the target object, such that the radius of the radius of the projectile is given as [FarinellaVokrouhlickyHartmann98]

$$r_{projectile} = \left(\frac{2\sqrt{2} R \omega \rho_{target}}{5 \rho_{projectile} V} \right)^{1/3} R$$

which, assuming $\rho_{target} = \rho_{projecile}$, simplifies to

$$r_{projectile} = \left(\frac{2\sqrt{2}\omega}{5V} \right)^{1/3} R^{4/3} \tag{7.1}$$

V is the typical collisional velocity in the asteroid belt (5000 m/s). The probability of a rotation rate reset through collision with a large enough projectile is given by

$$\frac{1}{\tau_{rot}} = P_i R^2 \cdot 3.5 \cdot 10^5 \cdot (r_{projectile})^{-5/2} \tag{7.2}$$

with _math:*P_i* the intrinsic collisional probability for the asteroid belt, $2.85 \cdot 10^{-18}\text{km}^2\text{yr}^{-1}$ [FarinellaVokrouhlicky-Hartmann98]. Using this number, and replacing $r_{projectile}$ in equation (7.2) with the expression in equation (7.1) results in a final probability of

$$\frac{1}{\tau_{rot}} = 1.0 \cdot 10^{-18} \cdot R^{-4/3} \left[\frac{2\sqrt{2}\omega}{5V} \right]^{-5/6}$$

per second (all values in SI units). If the rotation rate is reset, then both the obliquity and the new rotation rate are randomized.

7.1.3 Set the call interval

The two described probabilities of fragmentation and rotation reset events per time step, can have very small numbers, e.g. in the order of $1.0e-11$ or even lover. Using a random number generator to filter for such small numbers can be problematic. Therefore the calling interval of the two functions can be increased with the `Small Collisions Interval` option.

If this value is set to x, then the probabilities is multiplied by x, and the functions are called only at every x time steps. This also increases the performance of the code, since the functions are executed less times.

The values of literal:*Small Collisions Interval* should be chosen such that the probabilities p lie in

$$\text{def\_Rand\_Min} \ll p \ll 1$$

where `def_Rand_Min` is set in the `define.h` file (default = 1.0e-12).

CHAPTER

EIGHT

CREATE PARTICLE MODEL

8.1 Particles Creation Model

This model allows the creation of test particles during a running simulation. The model can be enabled with the `Create Particles file name` parameter in the *param.dat* file. If a file name is specified here, then the model is activated.

It is only available for test particles.
All particle creation events are reported in the Fragments file *The Fragments File: Fragments<name>.dat*.

The maximum number of particles in a simulation is the number of initial particles and test particles plus the value set in `Nfragments`.

$$N_{Maximum} = N(Initialbodies) + N(Initialtestparticles) + Nfragments$$

If the maximum number of allowed particles is reached, then no more new particles are created until some other particles are removed from the simulation by a collision or ejection event.

The following parameters are relevant for the particle creation model and can be set in the *param.dat* file:

- `Create Particles file name`, enable small bodies collisions model.
- `Nfragments`, Number of additional memory size for debris particles, in particle numbers, (default 0).

The first line in the 'Create Particles file' contains the particle creation mode:

- `Create Particles mode`:
  - 1: particles are generated by specified ranges in Kepler elements.
  - 2: particles are generated as escaping particles from specified parent bodies.

8.1.1 Particles Creation Mode 1

Enabled with `Create Particles mode = 1`.

The coordinates of the new generated particles are set by drawing uniform random numbers in a range of Kepler elements.

The particle creation file contains the following parameters:

- Create Particles mode = 1
- Maximum number of particles: Includes all particles, if this number is reached, then no new particles are generated
- Creation rate: in years, this specifies how frequent particles are generated.

- a: mean value of semi-major axis, in AU
- da: width of semi-major axis range, in AU
- e: mean value of eccentricity
- de: width of eccentricity range
- inc: mean value of inclination, in radians
- dinc: width of inclination range, in radians
- m: mass, in Solar Masses
- r: physical radius, in AU

The additional Keplerian elements (Ω, ω and M) are generated between 0 and 2π.

Example:

```
Create Particles mode = 1
Maximum number of particles = 100
Creation rate = 50.0
a = 5.2
da = 4.0
e = 0.8
de = 0.05
inc = 0.05
dinc = 0.05
m = 0.0
r = 1.0e-10
```

8.1.2 Particles Creation Mode 2

Enabled with `Create Particles mode = 2`.

The coordinates of the new generated particles are generated as escaping particles from parent bodies. A list of parent bodies can be specified in the file. New generated particles are placed 3 times the physical radius of the parent body away from its center. The direction is chosen isotropic and randomly. The escaping velocity is chosen randomly between the range Vmin and Vamx. Where Vmin and Vmax are given in terms of the escape velocity of the parent body.

For example: Vmin = 1 and Vmax = 2, then the escaping velocity of the new particle between 1 and 2 times $v_{esc} = \sqrt{2GM/d}$, where $d = 3R$, M and R are the mass and radius of the parent body.

The particle creation file contains the following parameters:

- Create Particles mode = 2
- Maximum number of particles: Includes all particles, if this number is reached, then no new particles are generated
- Creation rate: in years, this specifies how frequent particles are generated.
- Vmin, minimum velocity, in terms of the escape velocity of the parent body.
- Vmax, maximum velocity, in terms of the escape velocity of the parent body.
- m: mass, in Solar Masses
- r: physical radius, in AU
- List of particle indizes: In the following, the indizes of the parent bodies must be listed.

Example:

```
Create Particles mode = 2
Maximum number of particles = 100
Creation rate = 50.0
Vmin = 1.0
Vmax = 1.0
m = 0.0
r = 1.0e-10
List of particle indizes:
4
8
12
```

CHAPTER

NINE

BIBLIOGRAPHY

9.1 Bibliography

CHAPTER

TEN

INDICES AND TABLES

- genindex
- modindex
- search

BIBLIOGRAPHY

[BB00] Martin Beech and Peter Brown. Fireball flickering: the case for indirect measurement of meteoroid rotation rates. *Planetary and Space Science*, 48:925–932, August 2000. URL: http://adsabs.harvard.edu/abs/2000P%26SS...48..925B (visited on 2017-08-02), doi:10.1016/S0032-0633(00)00058-1.

[BRB00] William F. Bottke, David P. Rubincam, and Joseph A. Burns. Dynamical evolution of main belt meteoroids: numerical simulations incorporating planetary perturbations and yarkovsky thermal forces. *Icarus*, 145(2):301 – 331, 2000. URL: http://www.sciencedirect.com/science/article/pii/S0019103500963619, doi:http://dx.doi.org/10.1006/icar.2000.6361.

[LZJ95] Jer-Chyi Liou, Herbert A. Zook, and A.A. Jackson. Radiation pressure, poynting-robertson drag, and solar wind drag in the restricted three-body problem. *Icarus*, 116(1):186–201, 1995. URL: https://www.sciencedirect.com/science/article/pii/S0019103585711207, doi:https://doi.org/10.1006/icar.1995.1120.

[Moy03] T. D. Moyer. *Formulation for Observed and Computed Values of Deep Space Network Data Types for Navigation*. Wiley-Interscience, 2003.

[Moy71] T.D. Moyer. *Mathematical Formulation of the Double-Precision-Orbit-Determination-Program (DPODP)*. JPL technical report. Jet Propulsion Laboratory, California Institute of Technology, 1971. URL: https://books.google.ch/books?id=7P4tHAAACAAJ.

[NF91] Akiko Nakamura and Akira Fujiwara. Velocity distribution of fragments formed in a simulated collisional disruption. *Icarus*, 92:132–146, July 1991. URL: http://adsabs.harvard.edu/abs/1991Icar...92..132N (visited on 2017-07-31), doi:10.1016/0019-1035(91)90040-Z.

[Wie15] Paul A. Wiegert. Meteoroid impacts onto asteroids: A competitor for Yarkovsky and YORP. *Icarus*, 252:22–31, May 2015. URL: http://linkinghub.elsevier.com/retrieve/pii/S0019103514007131 (visited on 2016-12-16), doi:10.1016/j.icarus.2014.12.022.

[AuclairDesrotourLePLafitteMathis14] P. Auclair-Desrotour, C. Le Poncin-Lafitte, and S. Mathis. Impact of the frequency dependence of tidal Q on the evolution of planetary systems. *aap* , 561:L7, January 2014. arXiv:1311.4810, doi:10.1051/0004-6361/201322782.

[BolmontRaymondLeconte+15] E. Bolmont, S. N. Raymond, J. Leconte, F. Hersant, and A. C. M. Correia. Mercury-T: A new code to study tidally evolving multi-planet systems. Applications to Kepler-62. *aap* , 583:A116, November 2015. arXiv:1507.04751, doi:10.1051/0004-6361/201525909.

[BurnsLamySoter79] J. A. Burns, P. L. Lamy, and S. Soter. Radiation forces on small particles in the solar system. *icarus* , 40:1–48, October 1979. doi:10.1016/0019-1035(79)90050-2.

[Chambers99] J. E. Chambers. A hybrid symplectic integrator that permits close encounters between massive bodies. *mnras* , 304:793–799, April 1999. doi:10.1046/j.1365-8711.1999.02379.x.

[CorreiaLaskarFaragoBoue11] A. C. M. Correia, J. Laskar, F. Farago, and G. Boué. Tidal evolution of hierarchical and inclined systems. *Celestial Mechanics and Dynamical Astronomy*, 111:105–130, October 2011. arXiv:1107.0736, doi:10.1007/s10569-011-9368-9.

[Efroimsky07] Michael Efroimsky. Physics of bodily tides in terrestrial planets and the appropriate scales of dynamical evolution. *Journal of Geophysical Research (Planets)*, 112(E12):E12003, December 2007. arXiv:0709.1995, doi:10.1029/2007JE002908.

[Fabrycky10] D. C. Fabrycky. Non-Keplerian Dynamics. *ArXiv e-prints*, June 2010. arXiv:1006.3834.

[FarinellaVokrouhlickyHartmann98] P. Farinella, D. Vokrouhlický, and W. K. Hartmann. Meteorite Delivery via Yarkovsky Orbital Drift. *icarus* , 132:378–387, April 1998. doi:10.1006/icar.1997.5872.

[GrimmStadel14] Simon L. Grimm and Joachim G. Stadel. The GENGA Code: Gravitational Encounters in N-body Simulations with GPU Acceleration. *apj* , 796(1):23, November 2014. arXiv:1404.2324, doi:10.1088/0004-637X/796/1/23.

[GrimmStadelBrasser+22] Simon L. Grimm, Joachim G. Stadel, Ramon Brasser, Matthias M. M. Meier, and Christoph Mordasini. GENGA II: GPU planetary N-body simulations with non-Newtonian forces and high number of particles. *arXiv e-prints*, pages arXiv:2201.10058, January 2022. arXiv:2201.10058.

[HoffmannGrimmMooreStadel17] Volker Hoffmann, Simon L. Grimm, Ben Moore, and Joachim Stadel. Stochasticity and predictability in terrestrial planet formation. *mnras* , 465(2):2170–2188, February 2017. arXiv:1508.00917, doi:10.1093/mnras/stw2856.

[Hut81] P. Hut. Tidal evolution in close binary systems. *aap* , 99:126–140, June 1981.

[Kidder95] L. E. Kidder. Coalescing binary systems of compact objects to (post)$^5/2$-Newtonian order. V. Spin effects. *prd* , 52:821–847, July 1995. arXiv:gr-qc/9506022, doi:10.1103/PhysRevD.52.821.

[MardlingLin02] R. A. Mardling and D. N. C. Lin. Calculating the Tidal, Spin, and Dynamical Evolution of Extrasolar Planetary Systems. *apj* , 573:829–844, July 2002. doi:10.1086/340752.

[MorishimaStadelMoore10] Ryuji Morishima, Joachim Stadel, and Ben Moore. From planetesimals to terrestrial planets: N-body simulations including the effects of nebular gas and giant planets. *icarus* , 207(2):517–535, June 2010. arXiv:1007.0579, doi:10.1016/j.icarus.2009.11.038.

[SahaTremaine94] P. Saha and S. Tremaine. Long-term planetary integration with individual time steps. *aj* , 108:1962–1969, November 1994. arXiv:astro-ph/9403057, doi:10.1086/117210.

[VokrouhlickyFarinella99] D. Vokrouhlický and P. Farinella. The Yarkovsky Seasonal Effect on Asteroidal Fragments: A Nonlinearized Theory for Spherical Bodies. *aj* , 118:3049–3060, December 1999. doi:10.1086/301138.

[VokrouhlickyMilaniChesley00] D. Vokrouhlický, A. Milani, and S. R. Chesley. Yarkovsky Effect on Small Near-Earth Asteroids: Mathematical Formulation and Examples. *icarus* , 148:118–138, November 2000. doi:10.1006/icar.2000.6469.

[Yoshida90] Haruo Yoshida. Construction of higher order symplectic integrators. *Physics Letters A*, 150(5-7):262–268, November 1990. doi:10.1016/0375-9601(90)90092-3.

[ZdericMadigan20] Alexander Zderic and Ann-Marie Madigan. Giant-planet Influence on the Collective Gravity of a Primordial Scattered Disk. *aj* , 160(1):50, July 2020. arXiv:2004.00037, doi:10.3847/1538-3881/ab962f.