Documentation for the modified code Gadget–2

Demidova Tatiana

March 31, 2021

To work with the program code the package MPICH is need. The program code is compiled by a script

```
.\run.sh
```

and run by the command

```
mpirun -np N ./main.exe cbd.param
```

where N is number of processor cores for calculations.

1 Preprocessor directives

They are added as keys with the **-D** prefix to the *run.sh* file and are executed at compilation.

DIM1 — The calculations are carried out in a one-dimensional version and only the values of the x coordinate and the velocity v_x are used. In the file of initial conditions, the velocities for other spatial dimensions must be set to zeros. If the directive is not specified, then three-dimensional calculations are performed.

HSMLCONSTANT — In the calculations, a constant smoothing length is used, which is determined by the parameter **HsmlConstant** from parameter file *cbd.param* (see below). In this case, the values of the parameters **DesNumNgb** and **MaxNumNgbDeviation** are ignored. If the directive is not specified, the smoothing length is determined by the iteration method, so that near each particle the number of particles is within $DesNumNgb \pm MaxNumNgbDeviation$.

NOSTARSELFGRAV — It disables the effect of disk selfgravity on the motion of stars. The mass above which the influence of the selfgravity of the disk is not taken into account to calculate the stars motion is determined by the parameter **StarMassIndicator** from parameter file *cbd.param*.

ISOTHERM_EQS — this special option makes the gas behave like an isothermal gas with equation of state $P = c_s^2 \rho$. The sound speed c_s is set by the thermal energy per unit mass in the initial conditions, i.e. $c_s^2 = u$. If the value for u is zero, then the initial gas temperature **InitGasTemp** in the parameter file *cbd.param* is used to define the sound speed according to $c_s^2 = k\,T/m_p$, where m_p is the proton mass and k is Boltzmann constant.

SAVETEMP_LTR — this option is used in calculating the dynamics of protoplanetary disks in the approximation of local thermodynamic equilibrium while maintaining the temperature profile in the disk:

$$T(r,z) = \begin{cases} T_{mid} + (T_{atm} - T_{mid}) \left[\sin\left(\dfrac{\pi z}{2\chi H(r)}\right)\right]^4, & z < \chi H(r) \\ T_{atm}, & z > \chi H(r). \end{cases} \tag{1}$$

Where $T_{mid}(r) = \sqrt[4]{\phi}\left(\frac{R_*}{r}\right)^p T_*$ is the temperature in the disk plane and $T_{atm}(r) = T_*\left(\frac{R_*}{2r}\right)^q$ — the temperature of the disk atmosphere, and $H(r) = \sqrt{\frac{\kappa T_{mid}(r) r^3}{G M_* \mu m_H}}$ is disk half thickness. T_* and R_* are temperature and size of the object in the center of mass of the system are determined by parameters **CentralTemperature** and **CentralRadius** in the parameter file *cbd.param.* They are ignored when this option is disabled. M_* is the mass of center mass of the system. $\sqrt[4]{\phi}$ is defined by variable **Tcoef**, p — by **Pindex**, q — by **Aindex**, χ — **heightScale** — in the header file *allvars.h.* $\mu = 4.0/(1 + 3F_h)$, where hydrogen fraction in gas F_h is determined by **HYDROGEN_MASSFRAC** in the header file *allvars.h.* G is gravitational constant, m_H is mass of the hydrogen atom and k is Boltzmann constant.

If neither ISOTHERM_EQS option no SAVETEMP_LTR are added, then the adiabatic equation of state is used for gas $P = u(\gamma - 1)\rho$. u is the thermal energy per unit mass and γ is defined by variable **GAMMA** in the header file *allvars.h*

MKSKHEMA — if the option is specified, the algorithm from *J.J. Monaghan, A. Kocharyan*, Computer Physics Communications, **87**, 225 (1995) is used for the interaction of dust and gas particles. Else the algorithm from *O.P. Stoyanovskaya, T.A. Glushko, N.V. Snytnikov et al.*, Astronomy and Computing, **25**, 25 (2018) is used. If there is no dust or gas fraction, then the option is ignored.

DECART — this option is used in calculating the dynamics of dust and gas interaction. To calculate the number of dusty or gas neighbors, a uniform Cartesian grid is used with a step **DustGasMechStep** within **BarrierDistance** from parameter file *cbd.param.* Otherwise, a spherical mesh uniform along the radius with a step **DustGasMechStep** within **BarrierDistance** is used, the step along the angle ϕ is determined by the parameter **DustGasMechAngle**, along θ **DustGasMechAngle**.Ignored if preprocessor directive **MKSKHEMA** is specified.

2 List of the parameters in file cbd.param

InitCondFile — the path to the initial conditions file in binary format.

OutputDir — pathname of the output directory of the code.

RestartFile — basename of restart-files produced by the code.

SnapshotFileBase — basename of snapshot files produced by the code.

TimeBegin — this sets the starting time of a simulation when the code is started from initial conditions. In the restart mode this time should be equal to **TimeMax** (see below) of the previous calculation. The parameter is specified in system units which is defined by parameters **UnitVelocity_in_cm_per_s** and **UnitLength_in_cm**. Type is double.

TimeMax — this sets the final time for the simulation. In the restart mode this time should be change to a greater value. The parameter is specified in system units which is defined by parameters **UnitVelocity_in_cm_per_s** and **UnitLength_in_cm**. Type is double.

TimeBetSnapshot — the time interval between two subsequent snapshot files. The parameter is specified in system units which is defined by parameters **UnitVelocity_in_cm_per_s** and **UnitLength_in_cm**. Type is double.

TimeOfFirstSnapshot — the time of the first desired snapshot file. The parameter is specified in system units which is defined by parameters **UnitVelocity_in_cm_per_s** and **UnitLength_in_cm**. Type is double.

ErrTolIntAccuracy — this dimensionless parameter controls the accuracy of the timestep criterion selected by **TypeOfTimestepCriterion**.

CourantFac — this sets the value of the Courant parameter used in the determination of the hydrodynamical timestep of SPH particles. Type is double.

MaxSizeTimestep — this gives the maximum time step a particle may take. This should be set to a sensible value in order to protect against too large timesteps for particles with very small acceleration. The parameter is specified in system units which is defined by parameters **UnitVelocity_in_cm_per_s** and **UnitLength_in_cm**. Type is double.

MinSizeTimestep — this gives the minimum time step a particle may take. The parameter is specified in system units which is defined by parameters **UnitVelocity_in_cm_per_s** and **UnitLength_in_cm**. Type is double.

ErrTolTheta — this gives the maximum opening angle if the Barnes & Hut criterion is used for the tree walk. If the relative opening criterion is used instead, a first force estimate is computed using the Barnes & Hut algorithm, which is then recomputed with the relative opening criterion. Type is double.

TypeOfOpeningCriterion — this selects the type of cell-opening criterion used in the tree walks. A value of '0' results in standard Barnes & Hut, while '1' selects the relative opening criterion of GADGET-2. Type is double.

ErrTolForceAcc — the accuracy parameter for the relative opening criterion for the tree walk. Type is double.

TreeDomainUpdateFrequency — the domain decomposition and tree construction need not necessarily be done every single time step. Instead, tree nodes can be dynamically updated, which is faster. However, the tree walk will become more expensive since the tree nodes have to "grow" to keep accomodating all particles they enclose. The parameter controls how often the domain decomposition is carried out and the tree is reconstructed from scratch. For example, a value of '0.1' means that the domain decomposition and the tree are reconstructed whenever there have been more than $0.1N$ force computations since the last reconstruction, where N is the total particle number. A value of '0' will reconstruct the tree every time step. Type is double.

DesNumNgb — this sets the desired number of SPH smoothing neighbours. It is ignored if **HSMLCONSTANT** preprocessor directive used. Type is integer.

MaxNumNgbDeviation — this sets the allowed variation of the number of neighbours around the target value **DesNumNgb**. It is ignored if **HSMLCONSTANT** preprocessor directive used. Type is integer.

ArtBulkViscConst — this sets the value of the artificial viscosity parameter used by GADGET-2.

InitGasTemp — this sets the initial gas temperature in Kelvin when initial conditions are read. If this parameter is not equal to zero, the values stored in the initial condition file are ignored, otherwise the initial gas temperature is left at the value stored in the initial condition file. Type is double.

MinGasTemp — a minimum temperature in Kelvin floor imposed by the code. This may be set to zero. Type is double.

PartAllocFactor — each processor allocates space for the parameter times the average number of particles per processor. This number needs to be larger than '1' to allow the simulation to achieve a good work-load balancing, which requires to trade particle-load balance for work-load balance. It is good to make the parameter quite a bit larger than '1', but values in excess of '3' will typically not improve performance any more. For a value that is too small, the code may not be able to succeed in the domain decomposition and terminate. Type is double.

TreeAllocFactor — to construct the Barnes & Hut tree for N particles, somewhat less than N internal tree-nodes are necessary for 'normal' particle distributions. The parameter sets the number of internal tree-nodes allocated in units of the particle number. By experience, space for $0.65N$ internal nodes is usually fully sufficient, so a value of 0.7 should put you on the safe side. Type is double.

BufferSize — this specifies the size (in MByte per processor) of a communication buffer used by the code. Type is integer.

UnitVelocity_in_cm_per_s — this sets the internal velocity unit in cm/sec. Type is double.

UnitMass_in_g — this sets the internal mass unit in g. Type is double.

UnitLength_in_cm — this sets the internal length unit in cm. Type is double.

GravityConstantInternal — the numerical value of the gravitational constant G in internal units depends on the system of units you choose. For **GravityConstantInternal**=0, the code calculates the value corresponding to the physical value of G automatically. Type is double.

MinGasHsml — this parameter sets the minimum allowed SPH smoothing length. The smoothing length will be prevented from falling below this value. When this bound is actually reached, the number of smoothing neighbors will instead be increased above **DesNumNgb**. It is ignored if **HSMLCONSTANT** preprocessor directive used. Type is double.

HsmlConstant — the constant smoothing length for the gas and dust particles. It is ignored if **HSMLCONSTANT** preprocessor directive not used. Type is double.

SofteningStars — the Plummer equivalent gravitational softening length for star type particle. Type is double.

DustGasMechStep — the cells size for calculating the interaction of dust and gas. Ignored if preprocessor directive **MKSKHEMA** is specified. Type is double.

DustGasMechAngle — the angle step of ϕ in degrees for a spherical mesh for calculating the interaction of dust and gas. Ignored if preprocessor directive **MKSKHEMA** is specified. Type is double.

DustSize — dust particle size in cm. Type is double.

DustRho — dust density in g/cm^3. Type is double.

AccretionRadius — determines the size of the accretion radius for massive objects (stars and planets). If **AccretionRadius**=1, the accretion radius is defined as the distance at which a particle on a circular orbit travels 100**MinSizeTimestep**. Dust and gas particles approaching such an object at a distance closer than this radius are considered to have accreted onto the object and removed from the calculations. Type is double.

BarrierDistance — distance that defines the boundary of the computational domain from the center of mass of the system. Particles that have moved outside the boundaries are removed from the calculations. In the vertical direction, the boundary condition is determined by value

BarrierDistance/4. The parameter is specified in system units which is defined by parameters **UnitLength_in_cm**. Type is double.

StarMassIndicator — determines the minimum mass of an object in Solar masses that is a star. Used when a preprocessor directive **NOSTARSELFGRAV** is specified, otherwise the parameter is ignored. Objects with a mass greater than the specified mass are not affected by the self-gravity of the disk. Type is double.

CentralTemperature — temperature in Kelvin of matter disk in the distance **CentralRadius**. Used if the preprocessor **SAVETEMP_LTR** is specified, otherwise ignored. Type is double.

CentralRadius — The distance from the center of mass in the radii of the Sun at which **CentralTemperature** is set. Used if the preprocessor **SAVETEMP_LTR** is specified, otherwise ignored. Type is double.

3 Initial conditions

The path to the initial conditions file in binary format is defined by parameter **InitCondFile** in *cbd.param*. Three types of particles can be used in calculations: gaseous is type 0, dusty is type 1, and type 2 is stars. Either gas or dust particles must be present.

The program **InitCond/initial_Condition.c** create the initial condition file *InitCond/outbeg* from text files with gas, dust and stars particles parameters. The text file for gas particles parameters must to contain the coordinates, velocities, mass, internal energy per unit mass and id of the particles in format *(“%lf\t%lf\t%lf\t%lf\t%lf\t%lf\t%lf\t%lf\t%i\n”, x,y,z,vx,vy,vz,m, U,id)*. The same for dust and stars particles except internal energy per unit mass in format
(“%lf\t%lf\t%lf\t%lf\t%lf\t%lf\t%lf\t%i\n”, x, y, z, vx, vy, vz, m, id).

Examples of initial conditions are placed in folders *SODTest* and *StarsTest* for the Sod problem and modeling the protoplanetary disk of a binary system.

3.1 The Sod problem

The initial conditions for the problem are placed in folders **SODTest**. File **initialFiles.c** create the initial distribution of gas and dust particles in files **gas.in** and **dust.in**, respectively. The initial density, velocity, internal energy per unit mass defined as follows:

$$\begin{cases} \rho = 1, v = 0, U = 2.5, & x < 0, \\ \rho = 0.125, v = 0, U = 2, & x > 0. \end{cases} \tag{2}$$

The dust particles have the same density and velocity. 4800 gas and 4800 dust particles evenly distributed in the range $[-0.6 : 0.6]$. The particles with $|x| < 0.5$ simulate the border and don't move. The density drop is determined by the ratio of the masses of particles to the left (m_1) and to the right (m_2) of the discontinuity: $m_1/m_2 = 8$.

The parameter file **cbd.param** and script for compiling **run.sh** placed in folder **SODTest** should be copied to the code folder.

The folder **results** contains a program **read_snapshot.c** and script **read_snapshot.sh** for all files that converts the binary format into text file. It puts into folder **dat** files with the position, velocity, density, internal energy, pressure, particle type (0 gas, 1 dust) and ID.

3.2 The circumbinary disk

The initial conditions for the problem are placed in folder **StarsTest**. File **initialFiles.c** create the initial distribution of gas, dust and stars particles in files **gas.in**, **dust.in**, **stars.in** respectively. The binary masses, eccentricity, inclination, period and central temperature (T_*) (the last should be equal to parameter **CentralTemperature** in file *cbd.param*) are the input parameters. The $N_d \sim 5 \cdot 10^5$ dust and $N_g \sim 5 \cdot 10^5$ gas particles are distributed by law

$$\rho(r,z) = \frac{\Sigma_0}{\sqrt{2\pi}H(r)} \left(\frac{r}{r_0}\right)^{-1} e^{-\frac{z^2}{2H^2(r)}}, \tag{3}$$

where $\Sigma 0$ is determined by the numbers of particles N_g and N_d, $r_0 = 1.5$ semi-major axis, $H(r) = \sqrt{\frac{\kappa T_{mid}(r) r^3}{G M_* \mu m_H}}$. Here $\mu = 2.37$, M_* is sum of binary component mass, k is Boltzmann constant, G is gravity constant, m_H is hydrogen atom mass. $T(r,z) = T_{mid}(r) = \sqrt[4]{\phi}\left(\frac{R_*}{r}\right)^{0.5} T_*$, where $R_* = R_\odot$ and $\phi = 0.05$. All particles are put on the Keplerian orbits relative to the center of mass of the system. The mass ratio of the dust and gas particles is set 1 : 100 the same as in interstellar medium.

File **initialFiles.c** writes the input parameters of the problem in file **description.in**. The system of units from the file **UnitVelocity_in_cm_per_s, UnitMass_in_g, UnitLength_in_cm** should be puted in parameter file **cbd.param**.

The parameter file **cbd.param** and script for compiling **run.sh** placed in folder **StarsTest** should be copied to the code folder.

The folder **results** contains a program **read_snapshot.c** and script **read_snapshot.sh** for all files that converts the binary format into text file. It puts into folder **dat** files with positions x,y,z and id of particles. Prefix ‘g’ in file name for gas particles, ‘d’ for dust and ‘s’ for stars.