ssresample.py Documentation

Last Update: 7/18/18

Started By: Derek C. Richardson

Contact Info:
Department of Astronomy
University of Maryland
College Park MD 20742
Tel: 301-405-8786
E-mail: `dcr@astro.umd.edu`

Contents

1 OVERVIEW

This utility allows the user to replace one or more particles in an `ss` file each with the contents of an entire `ss` file scaled to fill the volume of the original particle(s). In this way, the original file is being "resampled" to increase the resolution in some or all regions.

2 USAGE

The simplest invocation of the utility is:

```
ssresample.py ssfile
```

where `ssfile` is the source file to be resampled. The full list of options is as follows:

`--help` If present, shows a usage message and exits.

`--keep` If present, directs the utility to keep all particles from the original `ss` file that are not replaced. This is useful if resampling is only being done on a portion of the `ss` file.

`--outfile OUTFILE` Specifies the name of the output `ss` file to contain the resampled data. Defaults to `resampled.ss` if omitted.

`--particles PARTICLES` Specifies the name of a text file listing the particles to be replaced. The file should consist of one or more particle order numbers (i.e., integers between 0 and $N-1$ inclusive, where N is the number of particles in the source file), one per line. For example, to replace only the 50th particle in the source file, the particle text file would consist of just the number 49 (recall counting starts at 0). The particle file is mandatory; if omitted, the name defaults to `particles.txt`. Optionally, `all` can be used in place of a filename to indicate that all particles should be replaced.

`--replacements REPLACEMENTS` Specifies the name of a text file listing filenames of replacement `ss` files. If the file consists of more than one `ss` filename, the replacement `ss` file to use will be selected randomly from the list for each particle replaced. The replacements file is mandatory; if omitted, the name defaults to `replacements.txt`.

`--sepmult SEPMULT` Specifies the separation multiplier to apply to the replacement particles (default 1.01, i.e., replacement particle separations will be increased by 1%). This option is provided as a way to adjust particle overlaps when using `pkdgrav`'s soft-sphere code, since the scaling of particle masses and radii will cause a change in the equilibrium overlap of self-gravitating particles in contact. To avoid any large repulsive forces being generated, the scaling should be set to compensate for the maximum expected overlap (1% in this example), but note that this will likely cause some or all particles to come out of contact, so a brief period of reequilibration will ensue.

Note the above options can be abbreviated to single letters, e.g., `-h` or `--h`, etc.

3 EXAMPLE

The `etc` folder in the `pkdgrav` code distribution contains a subfolder `resample` with the following items:

`README` Contains the invocation of `ssgen` used to generate the example source file.

`particles.txt` An example list of particles to replace in the source file.

`replacements.txt` An example list of replacements to use, in this case just a single `ss` file that was generated in the `sample` subfolder.

`sample` A subfolder with the following items:

- `demparams.log` The log output from `demparams` used to determine `dDelta` and `dKn` in `ss.par`.
- `mksample.py` A Python script that runs `ssgen` to generate a cluster of polydisperse particles, runs `pkdgrav` to settle the cluster under its own gravity, then runs `mkmov.py` to make a movie of the evolution. The final output from the `pkdgrav` simulation is used as the replacement example.
- `ss.par` `pkdgrav` parameter file for settling the cluster.
- `ssdraw.par` `ssdraw` parameter file for visualizing the cluster.

`ssdraw.par` `ssdraw` parameter file for visualizing `ssgen.ss` before replacement and for visualizing `resampled.ss` after replacement.

`ssgen.ss` Sample source `ss` file, generated using the parameters given in the `README` file.

To run the example, enter the `resample` subfolder and type `ssresample.py ssgen.ss`. This should generate `resampled.ss`, which can be examined with `ssinfo` or by running `ssdraw resampled.ss` and viewing the resulting `resampled.ras` file.

The `sample` subfolder is provided as an example automated way to generate your own equilibrated replacement clusters for future use. You will need to copy a `pkdgrav` binary compiled with `DEM`, fixed ball, and rotation dashpot enabled into this subfolder. Then type `./mksample.py`. If all goes well, after 5 to 10 minutes this will create `movie.mp4` (watch it with your favorite movie viewer) and will generate a new `ss.001000000`, which is the replacement file used in the example above. The `ssgen` parameters used to generate the file are specified near the beginning of the `mksample.py` script, but you would need to run `demparams` to update `dDelta` and `dKn` if you change these parameters.