User guide for

LIGER v1.0

by Mikolaj Borzyszkowski

Argelander Institute für Astronomie, Universität Bonn

LIGER (Light Cones with GEneral Relativity) is a tool to obtain the distribution of tracers (particles or galaxies) on the backward light-cone, given a sequence of outputs from a Newtonian N-body simulation. The code takes into account leading-order redshift-space distortions due to an inhomogeneous gravitational potential as derived from general relativity. LIGER is particularly suited to estimate galaxy correlations on large spatial scales.

Contents

0 Preamble & Disclaimer

To start with, please note that LIGER comes without any warranty, or guarantee to provide the correct result. LIGER is made publicly available under the GNU public licence (GPL)[1]. This means you are free to use, copy, distribute or modify the source code under the terms of GPL. If LIGER was useful for your scientific work, we kindly ask you to include the reference to the paper where the code is described (Borzyszkowski et al. 2017) in your publication.

1 Compilation

In the unpacked LIGER directory this documentation, a folder with the c-source and header files, a folder with parameter files and the makefile can be found. To compile the source use the makefile by running the `make` command. The code requires the FFTW[2] package to be installed. If FFTW is not installed in a standard directory you can specify the path of the library in the makefile. LIGER further uses the OpenMP library for multi-threading. However, the code can be compiled without the usage of OpenMP adding a flag in the makefile. It is though recommended to use multi-threading.

1.1 Compile-time options

Here, we list the options that can be set during the compilation of LIGER. Uncomment the corresponding line in the makefile and add the options to the `OPT` variable, if you want to include them.

`MEASURETIME` Using this option, an additional file will be written, that contains information on the wall-clock time needed in the different steps of the program.

`NOOPENMP` This compiles the code without the use of the OpenMP library. Then the program runs with a single thread only.

2 Running LIGER

The program is started by typing the command

```
./liger <ParameterFile>
```

All parameters necessary are supplied within the parameter file.

[1] http://www.gnu.org/licenses/
[2] http://fftw.org/

2.1 Run-time options

An example of a parameter file can be found in `ParameterFiles/ParaRun20_0`. In the following all parameters, needed by the code, are specified. Note that the details about the numerical simulations, like the cosmology, are read directly from the snapshots.

2.1.1 Input/Output

`DirOutput` Directory to write the output of the code.

`FileNameOutput` Name of the output file.

`DirGravPotential` Directory where to read/write the gravitational potential.

`IfWritePotential` Boolean value, whether the gravitational potential should be written to a file. Note that the code will still try to read the potential.

`DirSimulation` Full path to the simulation particles, from which the gravitational potential should be calculated.

`DirGalaxies` Full path to the observation targets, which the gravitational potential should be calculated. If the dark-matter particles should be shifted themselves the same path as `DirSimulation` should be supplied. Accross the different snapshots the order of the galaxies should be identical, i.e. in snapshot number five the galaxy at the 42nd position should correspond to the galaxy at the same position in all other snapshots.

`NumSimSnapshots` The number of snapshots available (used for `DirSimulation` and `DirGalaxies`). By default the highest snapshot number is the latest snapshot and the time of the observer.

2.1.2 Observer

`ObsPos` The 3D position of the observer in the same units as used in the snapshot. The time of the observer equals the latest snapshot supplied.

`Maxdist` Maximum distance of the final light cone, given as length in the same units as used in the snapshot.

`Mindist` Minimum distance of the final light cone, given as length in the same units as used in the snapshot.

2.1.3 Code options

`NGravGrid` The number of grid elements in one dimension for the calculation of the gravitational potential.

`NumThreads` The number of threads used.

`MemBufFactor` The code estimates the fraction of targets inside the light cone based on the assumption that the targets are uniformly distributed. Then this number is multiplied with `MemBufFactor` and the resulting number of elements is reserved for the output array. If during run-time there is more targets in the light cone then space reserved, the code aborts. Thus, for a highly clustered distribution of targets consider increasing the number.

`NTargetsInMem` Number of targets loaded by each thread into the working memory. Can be used to reduce the total memory consumption of the program.

3 File formats

3.1 Output files

At finish, LIGER writes four files of output in binary format,

`<DirOutput>/<FileNameOutput>`_realspace.dat ,
`<DirOutput>/<FileNameOutput>`_vRSD.dat ,
`<DirOutput>/<FileNameOutput>`_GRRSD.dat ,
`<DirOutput>/<FileNameOutput>`_magni.dat .

The first three files contain the position of the same galaxies on different light cones, while the last file contains the value of the magnification. The file appended with `GRRSD` contains the positions calculated with the full GR effects employed. Further, the file appended with `vRSD` uses only the radial peculiar velocity to shift the particles and the file `realspace` extracts the real-space light cone from the simulation (i.e. no shift is applied). Each of the three files starts with a float and an integer number, containing the maximum radial size of the light cone and the number of targets (`Ngal`) in the file, respectively. Following are the three coordinates for each particle of single precession float values. The centre of the coordinate system is placed at the observer. Each file contains the same number of targets, thus, some galaxies might included in the file which are slightly out of the required range. Furthermore, all galaxies are written in the same order, i.e. the n-th particle in the first file is originally the same as in the other files. The last file, appended with `magni`, contains the magnification value for the galaxies followed by the same float and integer number. The magnification values are written in three blocks of single precession floats of length `Ngal`. The first block of `Ngal` magnification values is the total magnification, while the second and third blocks are the convergence and Doppler part of the magnification, respectively (Borzyszkowski et al. 2017).

Further, LIGER has the option to write the gravitational potential for each snapshot `Snap` to the file

`<DirGravPotential>`/GravPot`<Snap>`_FullSize_Ng`<NGravGrid>`_cic.dat .

The file is written in binary format and contains an integer variable giving the number of grid elements `Ngrid` (along one axis) followed by `Ngrid`^{3} float numbers of the potential value. The 3D grid is witten as a 1D array, where the initial third dimension has the fastest varying index (row-major order).

If activated, LIGER writes the file

`<DirOutput>/<FileNameOutput>`_runtime.txt ,

which lists the wall-clock time spend in different parts of the code as simple text file. Beside the total execution time, the time spend in the two main parts (potential calculation and integration of the line of sight) of the code is listed.

3.2 Input files

Currently, only files in standard GADGET2 (Springel 2005) file format are supported. This format is convenient as it delivers all necessary information like the particle number of the simulation or the adopted cosmology. The given path to the simulation `DirSimulation` or the galaxies `DirGalaxies` is appended a three digit number to indicate the snapshot. Note that the order of the galaxies should by equal across all snapshots. Although the requirements on the file format seem quite strict, LIGER can be easily adapted to deal with any format.

3.2.1 Custom input format

There are four functions involved in reading the input data:

`src/load_snap.c:` `load_snapshot_header` ,
`src/load_snap.c:` `load_galaxies_header` ,
`src/load_snap.c:` `load_snapshot` ,
`src/load_snap.c:` `load_galaxies_partial` .

These are four placeholders. Each function builds a string indicating the location of the file needed and calls the function which reads the corresponding data. This should facilitate the usage of custom input file formats. Simply redirect these functions to your own implementation of the reading procedure.

The first two functions supply LIGER with the details about the number of galaxies, the particle number in the simulation, the background cosmology and the mass of each particle for each snapshot. They are called only during the first part of the code, where the potential is calculated. Currently, all matter particles (used to compute the potential) are restricted to have the same mass. It might be worth to allow for unequal particle mass in the future.

The latter two functions use the `struct particle_data` to return the position (and velocity) of the particles (or galaxies) to the main code. The function `load_snapshot` is used to read the position of the matter particles from the simulation for the calculation of the gravitational potential. The velocity information is unused here and the particle order is of no importance. The last function `load_galaxies_partial` reads the position

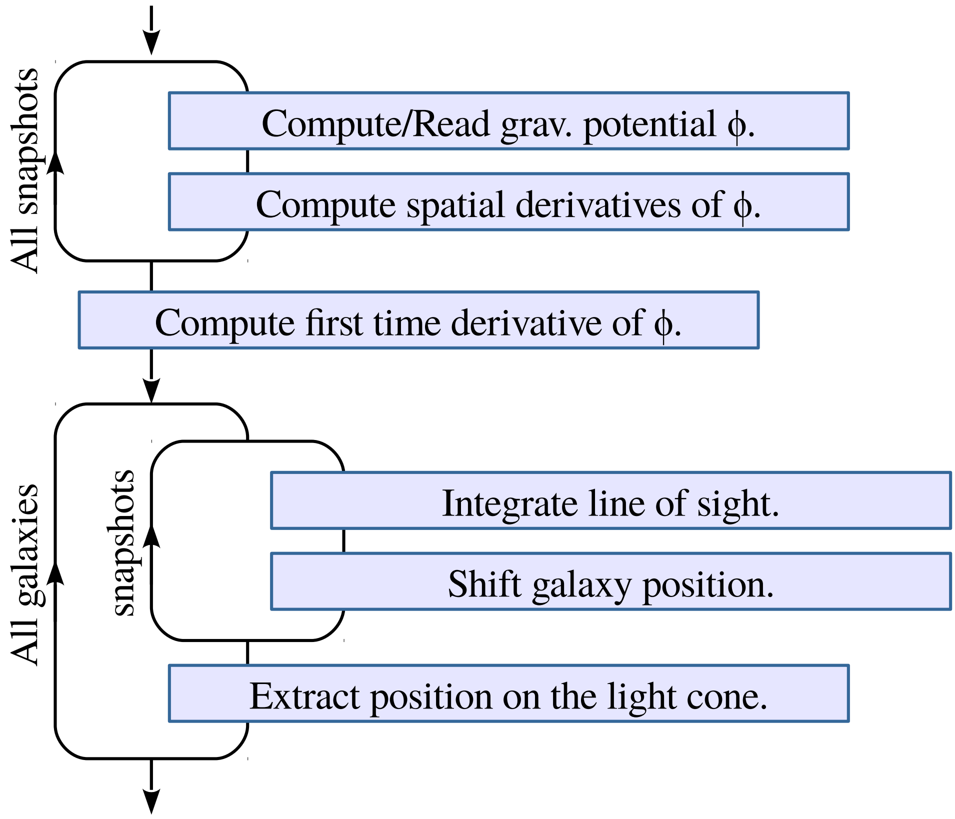

Figure 1: Schematic illustration of the structure of the code. The program runs from top to bottom. Closed lines symbolize the major loops in the code. Blue boxes show the major tasks performed by the code.

and velocities of the galaxies from a snapshot. These are the positions for which the shifts are computed and the position on the backward light cone is estimated. Note, however, that this function does not read the total number of targets, but only `NSelect` galaxies starting at index `NStart` (see `src/load_snap.h`). This is done to reduce the required memory of LIGER, since the position and velocity of each galaxy needs to be known at every snapshot. Here, it is essential that in the array returned by the function the galaxies have a consistent order between the different snapshots.

4 General code structure

LIGER calculates the distribution of galaxies on the light cone. We incorporate leading order GR effects from the perturbed metric to obtain the expected clustering properties on large-scales. The mathematical background has been presented in Borzyszkowski et al. (2017). The perturbation of the light cone can be taken into account by the coordinate transformation. The computation of the transformation involves an integral of the potential (and its derivatives) over the line of sight from the observer to the target.

First, LIGER computes the gravitational potential using the Poisson equation for each simulation snapshot (see Fig. 1). To estimate the potential, the simulation particles are used to build the density field on a regular grid using the Cloud-in-Cell method (Hockney and Eastwood 1988). Then the Poisson equation is solved in Fourier space to calculate the potential. LIGER has the option to write the potential to a file, such that the program can resort to the precomputed potential if another light cone for a different observer should be calculated from the same simulation. In the same loop, LIGER computes the spatial derivatives of the potential in Fourier space. After all snapshots have been processed, the code calculates the first time derivative of the potential using the finite-difference method.

Second, LIGER integrates the line of sight from the observer to each target position at different snapshots. The integration is performed by summing the values of all grid elements intersecting with the path, weighted by the length of the path inside the corresponding pixel (Amanatides and Woo 1987). The values of the grid are interpolated in time as a function of the distance to the observer, to follow the backward light cone in an unperturbed FLRW metric. Since the integral is the computationally most expensive part of the code, LIGER does not perform the integration to all snapshots but only the four snapshots around the expected crossing of the galaxy trajectory with the light-cone. In the end these four positions are shifted and the light-cone intersection is found by interpolating between these four snapshots. At the same time the magnification is calculated and interpolated similarly to the position.

The parallelisation scheme is chosen such that each thread gets `NTargetsInMem` number of targets and processes them independently from each other. The result is collected in a common array.

4.1 Memory requirements

During the full runtime of LIGER, the full grid of the gravitational potential and its derivatives are present in memory. The required memory depends on the used grid size and amounts to

$$M_\phi = \texttt{NGravGrid}^3 \times \texttt{NumSimSnapshots} \times 11 \times 8$$

bytes. Further, some memory for the final galaxy position is allocated. If the total number of galaxies is `NGal` and the size of the simulation box is `Box`, then the array size is

$$M_\mathrm{Out} = \texttt{MemBufFactor} \times \texttt{NGal} \times \left(\frac{4\pi(\texttt{Maxdist}^3 - \texttt{Mindist}^3)}{3\,\texttt{Box}^3} \right) \times 12 \times 8$$

bytes. The number of galaxies which are saved in memory during the computation of the coordinate transformation can be influenced with the `NTargetsInMem` parameter. The required memory is

$$M_\mathrm{Gal} = \texttt{NTargetsInMem} \times \texttt{NumThreads} \times \texttt{NumSimSnapshots} \times 6 \times 8$$

bytes. For the computation of the integral some additional memory is required, however, it is not saved permanently. Thus, the three most important contributions are M_ϕ, M_{Out} and M_{Gal}.

References

J. Amanatides and A. Woo. A Fast Voxel Traversal Algorithm for Ray Tracing. In *EG 1987-Technical Papers*. Eurographics Association, 1987. doi: 10.2312/egtp.19871000.

M. Borzyszkowski, D. Bertacca, and C. Porciani. LIGER: mock relativistic light-cones from Newtonian simulations. *ArXiv e-prints*, Mar. 2017.

R. W. Hockney and J. W. Eastwood. *Computer simulation using particles.* Adam Hilger, Bristol and Philadelphia, 1988.

V. Springel. The cosmological simulation code GADGET-2. MNRAS, 364:1105–1134, Dec. 2005.