

README

Julian B. Muñoz
(Dated: May 26, 2018)

These are the instructions to install and run RelicFast. Details of the code can be found in:

Efficient Computation of Galaxy Bias with Neutrinos and Other Relics, J. B. Muñoz and C. Dvorkin 2018 (<http://arxiv.org/abs/1805.11623>)

Please consider refering to that work if you use the code in a publication. This code is presented as-is, with no guarantees regarding its accuracy. Please report any bugs to julianmunoz@fas.harvard.edu.

I. INSTALLATION

All the necessary files are in the RelicFast folder, all source files are in the `source` folder and all headers in the `include` one.

The first thing you need is to install CLASS. For that, once you are in the RelicFast directory simply do:

```
>> cd CLASS_Current/  
>> make class
```

and CLASS will be installed in your computer¹. Note that you can use whatever version of CLASS you desire, by replacing the CLASS Current folder with your preferred version (and copying the explanatory base.ini file to the new folder). Do check that the columns of the transfer function outputs have NOT changed, since those are read by RelicFast.

If it comes to worst, you can use the code without the `Makefile`, simply by moving all files within `include` and `source` into the RelicFast folder, and running:

```
>> gcc -O4 -ffast-math -lstdc++ -fopenmp RelicFast.cpp bias.cpp collapse.cpp auxiliar.cpp  
>> boltzmann_Calls.cpp pressure.cpp -o relicfast
```

This is pretty inconvenient, which is why using the `Makefile` is the preferred option.

II. RUNNING RELICFAST

Once RelicFast is installed it's easy to run. If you are in the RelicFast folder you just have to type:

```
>> ./relicfast INPUT_FILE
```

¹ Visit <http://class-code.net/> if you encounter problems with CLASS.

for whatever inputs you put in `INPUT_FILE`. We provide the file example `input.ini` as an example, with instructions about the parameters.

Then, **RelicFast** will calculate the spherical collapse, and output the Lagrangian and Eulerian biases, as well as the matter and halo power spectra, for whatever redshifts z_{coll} of collapse are chosen, halo masses M , and wavenumbers k . This output will be saved to `output/result-X`, where **X** is a variable named `file_tag`. In the current version of **RelicFast** this variable is set simply by the characteristics of the light relic chosen, but this is easy to change (in the function `prepare cosmology` within `boltzmann.Calls.cpp`). Additionally, RelicFast will save an info file in the `result-X` folder, with the halo masses and redshifts you have ran, and will also copy the input file there.

Even though you probably do already, please refer to **CLASS** (1104.2933) or **CAMB** (astro-ph/9911177).

III. ADDITIONAL COMMENTS

There are some parameters that are not in the input file but you might want to vary (albeit not every run). Those would be in the `common.h` header, and would require recompiling the code.

In particular, if the code spits out a warning that the initial conditions are too narrow, you might have to change the initial conditions, for instance making `boost_initial_conditions` larger.

If you change the **CLASS/CAMB** version, you may have to adjust the `length_transfer` variable. (In addition, remember to copy the `explanatory_base.ini` to the new **CLASS** folder).

For large N_{eff} (ruled out by data), **CLASS** might not be able to solve for Y_{He} for BBN, so you would have to manually set Y_{He} in the **CLASS** input file.

There are two important precision parameters, precision scale controls how precisely the scale-dependence of $b_1^L(k)$ is calculated. `precision_normalization` controls the accuracy on the overall value of b_1^L , and is this less important.