CAMB/CLASS

Boltzmann Codes

CAMB

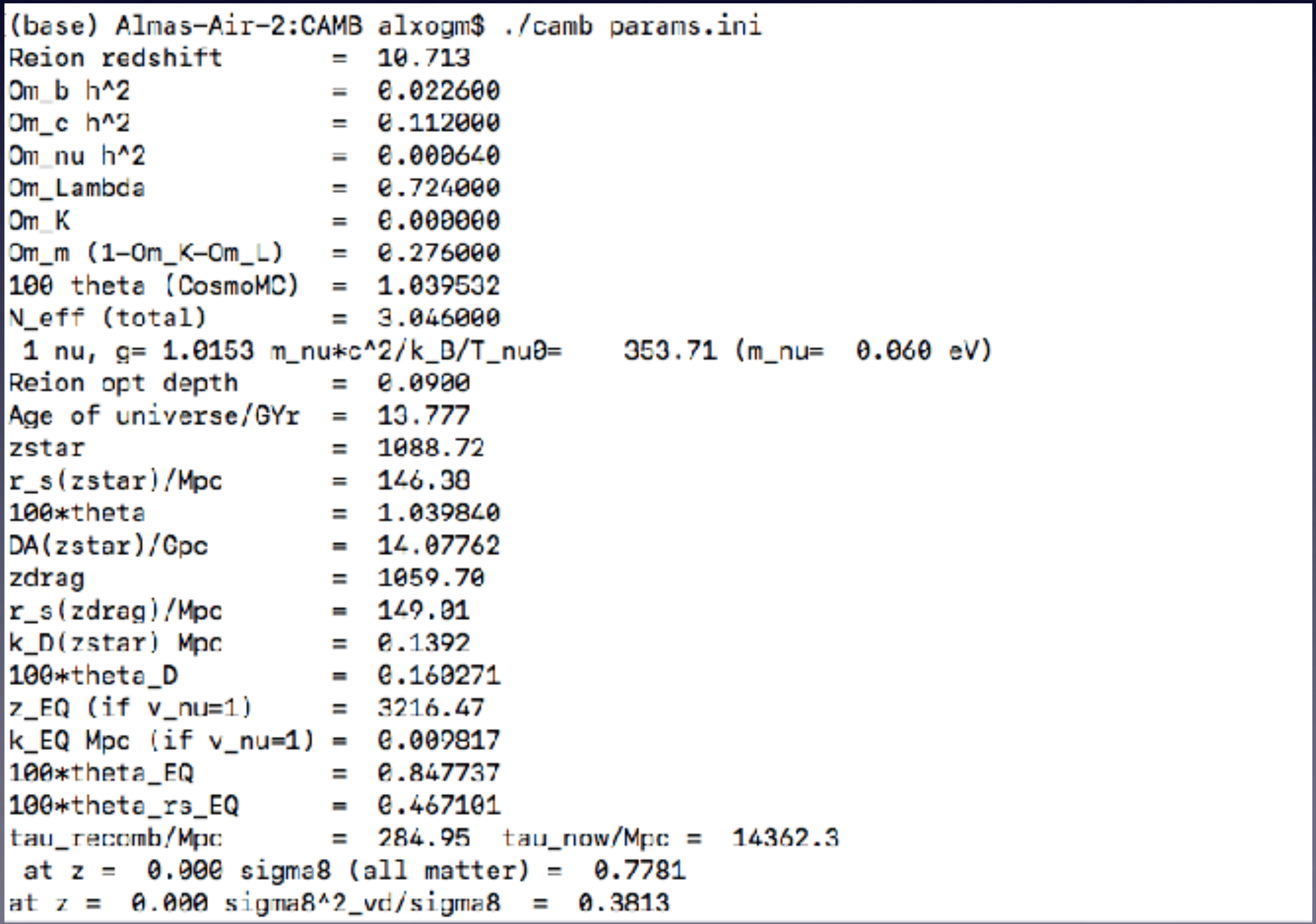
- https://camb.info
- https://github.com/cmbant/CAMB Includes python wrapper.
- Can be downloaded from any of those repositories.

Installation

- Needs a fortran compiler .
- Make file will check for ifort and gfortran, if any of those is present then installing is as simple as typing: make
  - tip: You can get a license of intel compilers for free with the student status. Or a 30 days trial.
  - gfortran 5 or higher should work.

Run camb from terminal

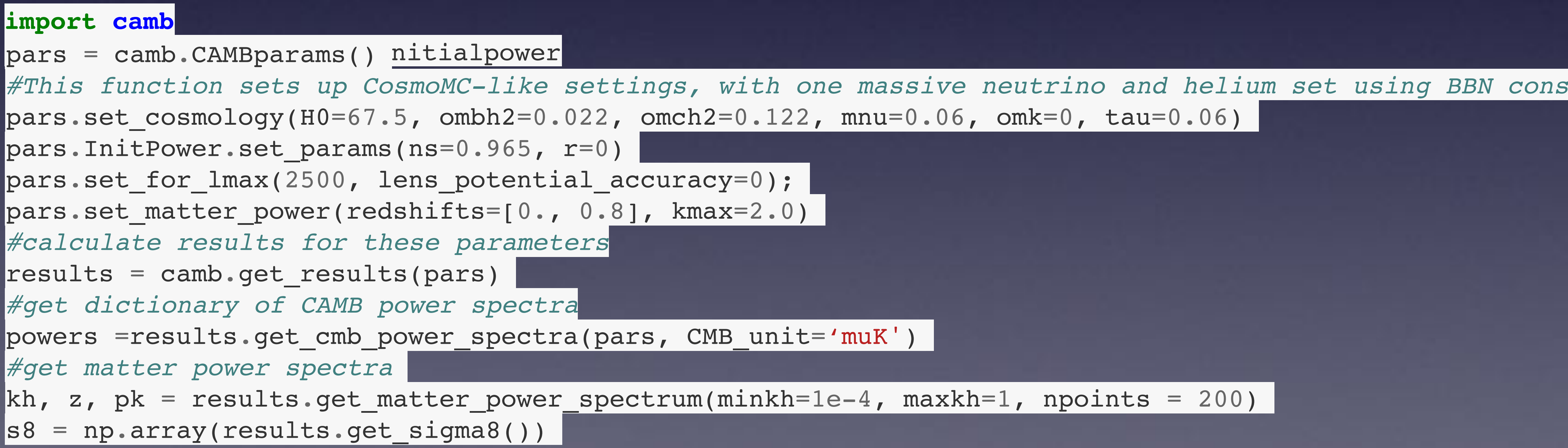
- Copy the params.ini to another file and call it params_or.ini so that you never loose the original , modify the params.ini according to your needs and run: ./camb params.ini



```
(base) Almas-Air-2:CAMB alxogm$ ./camb params.ini
Reion redshift       =  10.713
Om_b h^2             =  0.022600
Om_c h^2             =  0.112000
Om_nu h^2            =  0.000640
Om_Lambda            =  0.724000
Om_K                 =  0.000000
Om_m (1-Om_K-Om_L)   =  0.276000
100 theta (CosmoMC)  =  1.039532
N_eff (total)        =  3.046000
 1 nu, g= 1.0153 m_nu*c^2/k_B/T_nu0=    353.71 (m_nu=  0.060 eV)
Reion opt depth      =  0.0900
Age of universe/GYr  =  13.777
zstar                =  1088.72
r_s(zstar)/Mpc       =  146.38
100*theta            =  1.039840
DA(zstar)/Gpc        =  14.07762
zdrag                =  1059.70
r_s(zdrag)/Mpc       =  149.01
k_D(zstar) Mpc       =  0.1392
100*theta_D          =  0.160271
z_EQ (if v_nu=1)     =  3216.47
k_EQ Mpc (if v_nu=1) =  0.009817
100*theta_EQ         =  0.847737
100*theta_rs_EQ      =  0.467101
tau_recomb/Mpc       =  284.95  tau_now/Mpc =  14362.3
 at z =  0.000 sigma8 (all matter) =  0.7781
at z =  0.000 sigma8^2_vd/sigma8  =  0.3813
```

PYCAMB

- Go to the pycamb directory and type: python setup.py install
- Notebook example: http://camb.readthedocs.io/en/latest/CAMBdemo.html
  - Quite complete, and works nice. Example lines:



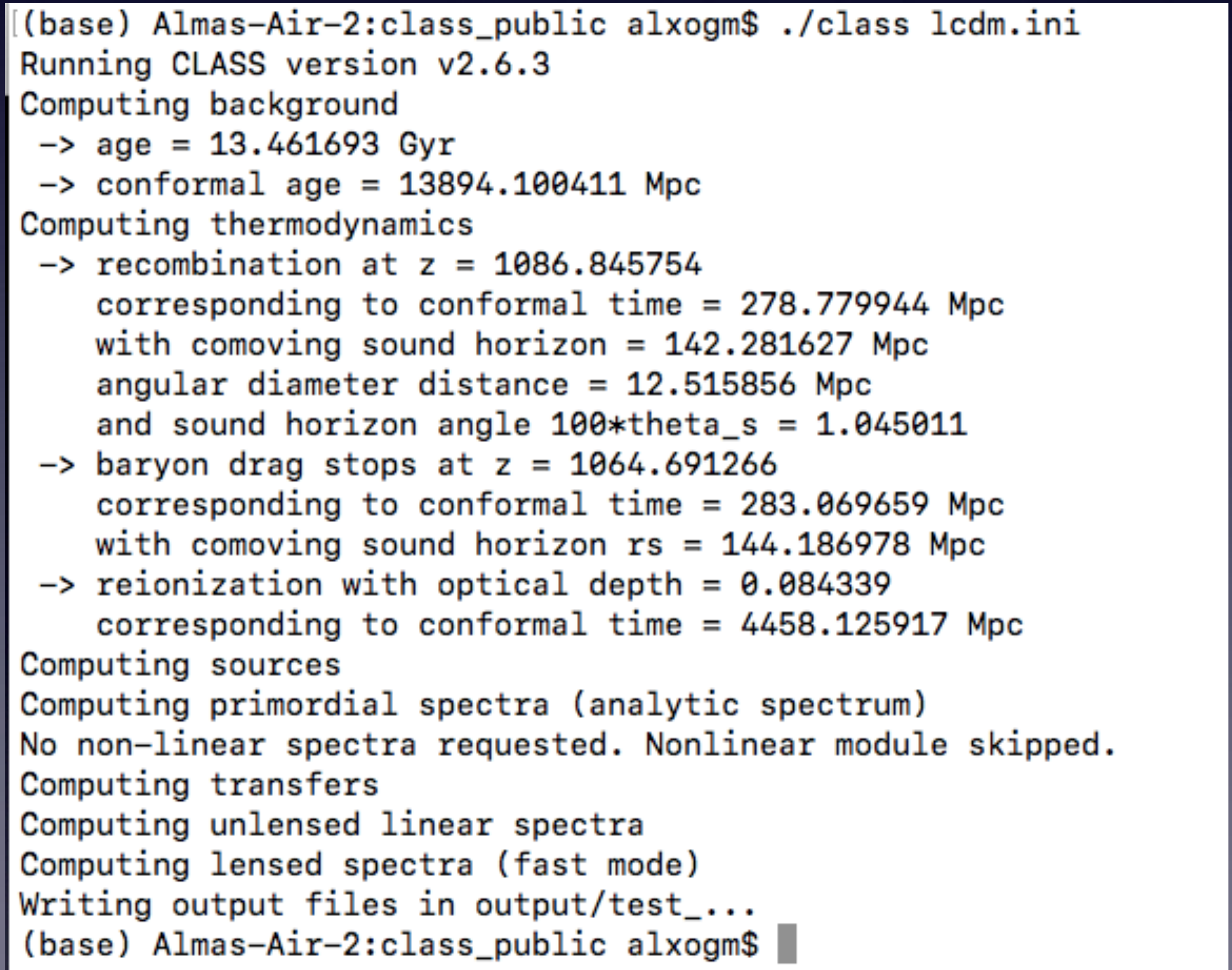
```
import camb
pars = camb.CAMBparams() nitialpower
#This function sets up CosmoMC-like settings, with one massive neutrino and helium set using BBN cons
pars.set_cosmology(H0=67.5, ombh2=0.022, omch2=0.122, mnu=0.06, omk=0, tau=0.06)
pars.InitPower.set_params(ns=0.965, r=0)
pars.set_for_lmax(2500, lens_potential_accuracy=0);
pars.set_matter_power(redshifts=[0., 0.8], kmax=2.0)
#calculate results for these parameters
results = camb.get_results(pars)
#get dictionary of CAMB power spectra
powers =results.get_cmb_power_spectra(pars, CMB_unit=‘muK')
#get matter power spectra
kh, z, pk = results.get_matter_power_spectrum(minkh=1e-4, maxkh=1, npoints = 200)
s8 = np.array(results.get_sigma8())
```

CLASS

- Lots of information and lecture notes in http://class-code.net
- Clone from to have the latest https://github.com/lesgourg/class_public
- Needs a C compiler: icc or gcc. gcc version 5 and higher works best.

Run class from terminal

- Copy the lcdm.ini (and/or the explanatory.ini) to another file and call it lcdm_or.ini so that you never loose the original , modify the lcdm.ini (or the explanatory.ini) according to your needs and run: ./class params.ini



```
[(base) Almas-Air-2:class_public alxogm$ ./class lcdm.ini
Running CLASS version v2.6.3
Computing background
 -> age = 13.461693 Gyr
 -> conformal age = 13894.100411 Mpc
Computing thermodynamics
 -> recombination at z = 1086.845754
    corresponding to conformal time = 278.779944 Mpc
    with comoving sound horizon = 142.281627 Mpc
    angular diameter distance = 12.515856 Mpc
    and sound horizon angle 100*theta_s = 1.045011
 -> baryon drag stops at z = 1064.691266
    corresponding to conformal time = 283.069659 Mpc
    with comoving sound horizon rs = 144.186978 Mpc
 -> reionization with optical depth = 0.084339
    corresponding to conformal time = 4458.125917 Mpc
Computing sources
Computing primordial spectra (analytic spectrum)
No non-linear spectra requested. Nonlinear module skipped.
Computing transfers
Computing unlensed linear spectra
Computing lensed spectra (fast mode)
Writing output files in output/test_...
(base) Almas-Air-2:class_public alxogm$ ▌
```

CLASS Python

- Needs cython if want to compile with the python wrapper.
- “make” will automatically install the classy object.
- “make class” , will only compile class but no the python wraper.
- If you will be dealing with different versions of the class code (e.g class.FreeSF https://github.com/lurena-lopez/class.FreeSF) It can be useful to rename the classy_obj in python/setup.py and compile again.

See terminal and Notebook

https://github.com/olegs22/TallerCosmoProject/blob/master/notebooks/CAMB_CLASS.ipynb