CCA New York workshop: CLASS Exercises

Julien Lesgourgues & Thomas Tram
lesgourg@physik.rwth-aachen.de

August 23, 2019

1 Basic usage: input syntax, standard output

1.1 Playing with the Planck best-fit model

There is no advantage in doing this exercise in a python script/notebook, because it requires no plotting, nor any manipulation of input/output data... So you can just run in a terminal with an input file.

Here is the table of best-fit parameters from the Planck 2013 Cosmological Parameter paper:

	Planck		*Planck*+lensing		*Planck*+WP	
Parameter	Best fit	68% limits	Best fit	68% limits	Best fit	68% limits
$\Omega_b h^2$	0.022068	0.02207 ± 0.00033	0.022242	0.02217 ± 0.00033	0.022032	0.02205 ± 0.00028
$\Omega_c h^2$	0.12029	0.1196 ± 0.0031	0.11805	0.1186 ± 0.0031	0.12038	0.1199 ± 0.0027
$100\theta_{MC}$	1.04122	1.04132 ± 0.00068	1.04150	1.04141 ± 0.00067	1.04119	1.04131 ± 0.00063
τ	0.0925	0.097 ± 0.038	0.0949	0.089 ± 0.032	0.0925	$0.089^{+0.012}_{-0.014}$
n_s	0.9624	0.9616 ± 0.0094	0.9675	0.9635 ± 0.0094	0.9619	0.9603 ± 0.0073
$\ln(10^{10} A_s)$	3.098	3.103 ± 0.072	3.098	3.085 ± 0.057	3.0980	$3.089^{+0.024}_{-0.027}$
Ω_Λ	0.6825	0.686 ± 0.020	0.6964	0.693 ± 0.019	0.6817	$0.685^{+0.018}_{-0.016}$
Ω_m	0.3175	0.314 ± 0.020	0.3036	0.307 ± 0.019	0.3183	$0.315^{+0.016}_{-0.018}$
σ_8	0.8344	0.834 ± 0.027	0.8285	0.823 ± 0.018	0.8347	0.829 ± 0.012
z_{re}	11.35	$11.4^{+4.0}_{-2.8}$	11.45	$10.8^{+3.1}_{-2.5}$	11.37	11.1 ± 1.1
H_0	67.11	67.4 ± 1.4	68.14	67.9 ± 1.5	67.04	67.3 ± 1.2
$10^9 A_s$	2.215	2.23 ± 0.16	2.215	$2.19^{+0.12}_{-0.14}$	2.215	$2.196^{+0.051}_{-0.060}$
$\Omega_m h^2$	0.14300	0.1423 ± 0.0029	0.14094	0.1414 ± 0.0029	0.14305	0.1426 ± 0.0025
$\Omega_m h^3$	0.09597	0.09590 ± 0.00059	0.09603	0.09593 ± 0.00058	0.09591	0.09589 ± 0.00057
Y_P	0.247710	0.24771 ± 0.00014	0.247785	0.24775 ± 0.00014	0.247695	0.24770 ± 0.00012
Age/Gyr	13.819	13.813 ± 0.058	13.784	13.796 ± 0.058	13.8242	13.817 ± 0.048
z_*	1090.43	1090.37 ± 0.65	1090.01	1090.16 ± 0.65	1090.48	1090.43 ± 0.54
r_*	144.58	144.75 ± 0.66	145.02	144.96 ± 0.66	144.58	144.71 ± 0.60
$100\theta_*$	1.04139	1.04148 ± 0.00066	1.04164	1.04156 ± 0.00066	1.04136	1.04147 ± 0.00062
z_{drag}	1059.32	1059.29 ± 0.65	1059.59	1059.43 ± 0.64	1059.25	1059.25 ± 0.58
r_{drag}	147.34	147.53 ± 0.64	147.74	147.70 ± 0.63	147.36	147.49 ± 0.59
k_D	0.14026	0.14007 ± 0.00064	0.13998	0.13996 ± 0.00062	0.14022	0.14009 ± 0.00063
$100\theta_D$	0.161332	0.16137 ± 0.00037	0.161196	0.16129 ± 0.00036	0.161375	0.16140 ± 0.00034
z_{eq}	3402	3386 ± 69	3352	3362 ± 69	3403	3391 ± 60
$100\theta_{eq}$	0.8128	0.816 ± 0.013	0.8224	0.821 ± 0.013	0.8125	0.815 ± 0.011
$r_{drag}/D_V(0.57)$	0.07130	0.0716 ± 0.0011	0.07207	0.0719 ± 0.0011	0.07126	0.07147 ± 0.00091

We will focus only on the best-fit ΛCDM model of Planck+WP (penultimate column).

1. Write a short input file with the appropriate values of ω_b, ω_{cdm} ($\Omega_c h^2$ in the paper), H_0 or h, τ_{reio} (τ in the paper). Run only the background and thermodynamics module for this model (to do so, just leave the `output` field blank, which is the default)[1]. To get some standard output, set `input_verbose`, `background_verbose` and `thermodynamics_verbose` to one. To be sure that you don't have syntax errors in your input file, setting `write warnings = yes` is healthy. Look at the different lines of standard output. Do you reproduce accurately the age of the Universe in Gyr given in the paper (here, accurately means e.g. up to better than 0.1σ) ?

2. Part of the difference comes from different settings for other parameters. To work with *exactly* the same parameters as in the Planck paper, you need to add one massive neutrino species with $m = 0.06$ eV. Thus you need to add one non-cold-dark-matter (ncdm) species, and to reduce the number of ultra-relativistic (ur) relics to two (plus small corrections coming from the details of neutrino decoupling models, which would be 0.046 for 3 massless neutrinos and 0.03351 for two massless neutrinos):

   ```
   N_ur = 2.03351
   N_ncdm = 1
   m_ncdm = 0.06
   ```

   Run again with these additional parameters. Do you get closer to the age indicated in the paper? Check also that you get the same or nearly the same value for the redshift and comoving sound horizon at recombination (z_*, r_* in the paper), redshift and comoving sound horizon at baryon drag (z_{drag}, r_{drag} in the paper), and reionization redhsift (z_{re} in the paper).

3. The tiny residual difference in the 6th digit of the age could be due either to precision settings or systematic errors in CAMB (used in the table) or CLASS. In CLASS there are only two parameters controlling the precision of the background integration. Here they are, with their default settings as implemented in `input.c`:

   ```
   back_integration_stepsize = 7.e-3
   tol_background_integration = 1.e-2
   tol_ncdm_bg = 1.e-5
   ```

   Try smaller values of these parameters (either in the same `.ini` file or in a `.pre` file, as you prefer), to check whether the calculation of the age at the level of the 6th digit is well converged or not with default settings.

4. Check that if instead of `h = 0.6704`, you pass `100*theta_s` with the value indicated in the Planck table (in the 3rd line), you get a different value of H_0. The reason is that the paper gives θ_{MC}, an analytic approximation to the actual ratio $\theta_s = d_s^{dec}/d_A^{dec}$ (sound horizon at decoupling over angular diameter distance to decoupling) computed internally by CosmoMC. Instead, CLASS computes the true $\theta_s = d_s^{dec}/d_A^{dec}$ with a numerical integration. To know the true θ_s, go back to the previous run in which you specified $h = 0.6704$ in input. Use the standard output of the code to get the correct $100\,\theta_s$. Now pass this value and check that you get the right h.

[1] in this exercise, we care only about the background and thermodynamics evolution, so it is not necessary to pass the same values of $\ln(10^{10}A_s)$ (or A_s) and n_s as in the paper. So they can be left unspecified. If we wanted to reproduce the same C_l's or $P(k)$ as for the Planck best-fit model, we would need to pass the correct $\ln(10^{10}A_s)$ (or A_s) and n_s, the correct pivot scale `k_pivot = 0.05` , and to specify some fields for `output` = ...

2 Python notebooks (or scripts)

2.1 Playing with output spectra

For the default ΛCDM model:

1. Plot on the same figure (log-lin) the lensed and unlensed C_l^{TT} of scalars, to see the effect of smoothing of the maxima and minima of the spectrum. When plotting inside the notebook with `%matplotlib notebook` you may find it nice to use the *zooming to rectangle* tool to zoom on peak maxima and minima.

2. Turn on tensor modes with a tensor-to-scalar ratio $r = 0.1$ (close to the current upper bound). Plot on the same figure (log-log) the contribution to C_l^{BB} from tensor modes and from lensing (or in other words, the unlensed C_l^{BB} and the difference between the lensed and unlensed ones). Check that B modes are dominated by lensing on small scales. The calculation goes much faster if you don't try to get the primordial C_l^{BB} up to very high l (the default, `l_max_tensors = 500`, is sufficient).

3. Plot on the same figure (log-log) the linear and non-linear matter power spectrum at $z = 0$ and $z = 2$, to see that at low redshift non-linear corrections appear on larger scales / smaller wavenumber.

2.2 Playing with thermodynamics

Plot the free electron fraction x_e as a function of redshift z for the default ΛCDM model, in log-log scale, for $0.1 < z < z_{\rm max}$, where $z_{\rm max}$ is returned by CLASS. Check visually that $x_e \ll 1$ between reionization ($z \sim 10$) and recombination ($z \sim 1100$).

2.3 Playing with inflation and primordial spectra

Here the goal is to use the inflation simulator and to compute the primordial scalar and tensor power spectrum for a quadratic inflation model, with a potential $V(\phi) = \frac{1}{2}m^2\phi^2$ and $\frac{1}{2}m^2 = 1.72 \times 10^{-12}$ (in natural units). After plotting the two spectra on the same figure (log-log), you can check that the values of A_s, n_s, α_s inferred by CLASS from the numerically computed primordial spectra look reasonable given Planck data, while r is a bit too high (this is why the model is now excluded at a few sigma's).

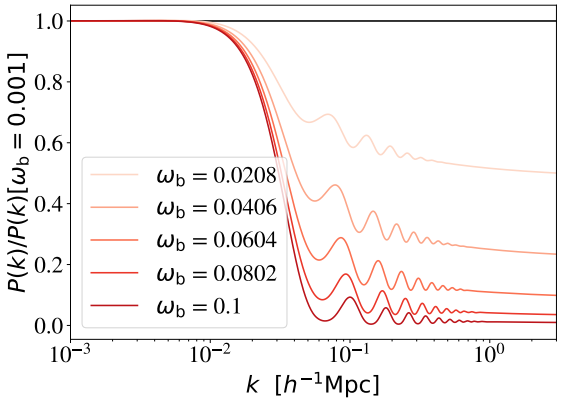
Hints: turn on the inflation simulator with `P_k_ini_type` set to `inflation_V_end` (this will only be accepted if you are computing some perturbations, thus `'output'` should not be empty, and if you require both scalar and tensor modes). Then, by default, the code will assume a polynomial potential of the form $V(\phi) = \sum_{i=0,\ldots,4} {\rm Vparam}_i \phi^i$. To switch on just the quadratic term, set `Vparam0` to `Vparam4` to zero, excepted `Vparam2 = 1.72e-12`. The number of e-folds between "horizon crossing for the pivot scale" and the end of inflation can be adjusted to 55 with the parameter `N_star`. The values of the derived parameters can be read either in the python notebook using the function `.get_derived_parameters(['A_s',...])`, or directly in the standard output if you have set `primordial_verbose` to 2.

2.4 Impact of baryons on the matter power spectrum

Plot the ratio of matter power spectrum for ~ 5 values of ω_b (log-lin) ranging from ~ 0 to 0.1, and check the effect of baryons: it should be a step-like suppression with superimposed oscillations (the BAOs).

Hints: Start from the public notebook `varying_neff.ipynb` and adapt it to your needs. Ideally you would take a reference model with $\omega_{\rm b} = 0$, but no Boltzmann code will allow you to do that: a certain amount of baryons is needed in order to have a tightly-coupled photon-baryon fluid at the time of initial conditions. Thus it is recommend to take a very small value like $\omega_{\rm b} = 10^{-3}$ as the reference. Since for the purpose of the plot you will use very unrealistic values of $\omega_{\rm b}$, the code cannot use its feature

of "automatically determining the primordial Helium fraction $Y_{\rm He}$ using an interpolation table encoding the results of BBN codes". Thus it is better to fix the primordial Helium fraction manually (syntax in CLASS python scripts: `'YHe':0.25`).



2.5 More advanced: visualizing the evolution of CMB perturbations for a few modes

Reproduce figure 8.1 from Scott Dodelson's book *Modern Cosmology*. The vertical axis is the combination of perturbations $k^{3/2}\,(\Theta_0 + \Psi)$, while the horizontal axis is the conformal time divided by the recombination time. This plot is pedagogically interesting because the CMB spectrum is given mainly by the value of this quantity evaluated at the time of recombination for different wavenumbers.

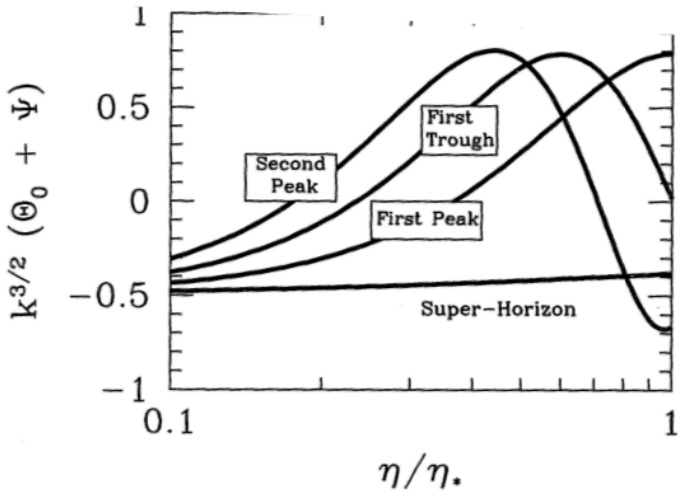


Figure 8.1. Evolution of photon perturbations of four different modes before recombination at η_*. Normalization is arbitrary, but the relative normalization of the 4 curves is appropriate for perturbations with a Harrison–Zel'dovich–Peebles ($n = 1$) spectrum. Model is standard CDM with $h = 0.5$, $\Omega_m = 1$, and $\Omega_b = 0.06$. Starting from the bottom left and moving upward, the wavenumbers for the modes are $k = (7 \times 10^{-4}, 0.022, 0.034, 0.045)$ Mpc^{-1} or $(8, 260, 400, 540)/\eta_0$.

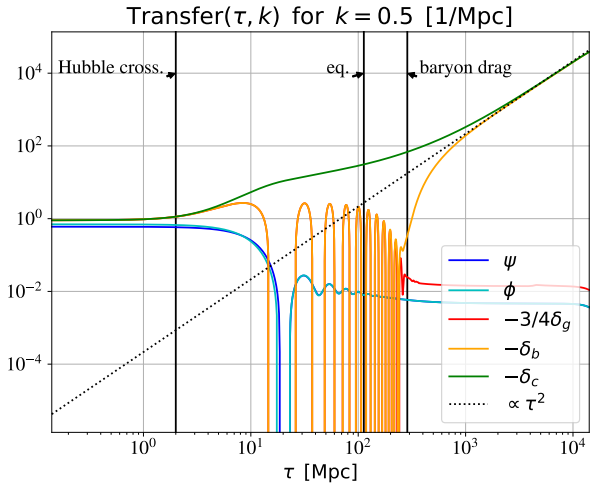
Hints: Given that $\Theta_0 \equiv \langle \delta T/T \rangle$ and that $\delta_\gamma \equiv 4\langle \delta T/T \rangle$, the quantity $k^{3/2}\,(\Theta_0 + \Psi)$ is given in CLASS

notations by $-2\left(\frac{1}{4}\texttt{delta\_g}+\texttt{psi}\right)$. The factor $k^{3/2}$ has been replaced with a -2 because CLASS normalizes all perturbations to an initial curvature perturbation $\mathcal{R}=1$ while Dodelson normalizes them to $\mathcal{R}=-\frac{1}{2}k^{-3/2}$. The ration η/η_* is given in CLASS notations by $\tau/\tau_{\rm rec}$, and the conformal time at recombination can be extracted with `.get_derived_parameters(['tau_rec'])`

2.6 More advanced: visualizing the evolution of all perturbations for one mode

Plot the evolution of the perturbations $-\delta_{\rm c}$, $-\delta_{\rm b}$, $-\frac{3}{4}\delta_\gamma$, ϕ, ψ for the wavenumber $k=0.5\,{\rm Mpc}^{-1}$, as a function of conformal time. Once you are done it can be fun to check a few well-known results from your cosmological perturbation courses:

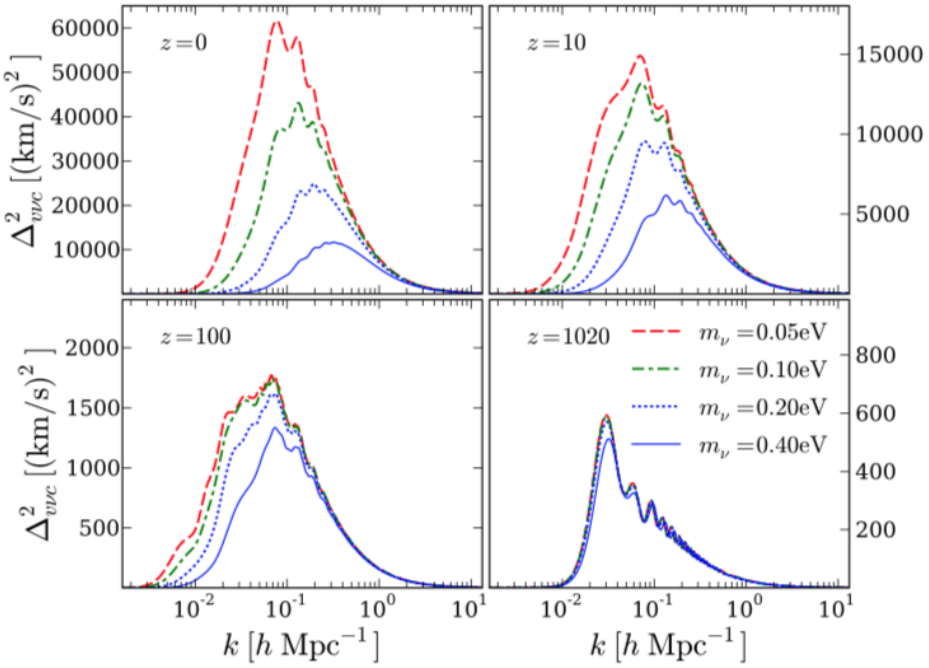
- ϕ and ψ nearly equal.
- $\delta_{\rm b}=-\frac{3}{4}\delta_\gamma$ until approaching the baryon drag time, and $|\delta_b|$ starting to grow after that time.
- there are five regimes for $\delta_{\rm c}$: constant, logarithmic growth (concave function), power-law growth going asymptotically like τ^2, small slow-down at the end (Λ domination).
- baryons try to catch up CDM during matter domination, but this is a slow process.



Hints: If you want to show like in the above figure the time of Hubble crossing (optional), you can simply approximate it as $\tau_*\simeq 1/k$. If you want to show the time of baryon drag (optional), it is easy to extract it with `.get_current_derived_parameters(['tau_d'])`. If you want to show the time of equality (optional), it is more difficult, because CLASS does not compute it. One way is to use `.get_background()` to build an array of values of $[(\omega_{\rm b}+\omega_{\rm cdm})/(\omega_\gamma+\omega_\nu)]$ versus time. Then the `interp1d` interpolation function can allow you to find the precise time at which this ratio crosses one. All this can be done in only 5 lines in the script.

2.7 More advanced: visualizing the transfer functions at a given time

Reproduce the following plot from Ue-Li Pen et al. 1311.3422, illustrating the contribution of various wavenumbers to the relative bulk flow between massive neutrinos and cold dark matter (in the synchronous gauge, which is the default in CLASS).



Hints: In CLASS notations,

$$\Delta^2_{\nu\nu c} \equiv \mathcal{P}(k) \left[\frac{\theta_\nu(k,z) - \theta_c(k,z)}{k}\right]^2$$

with $\mathcal{P}(k) = A_s(k/0.05)^{n_s-1}$, $\theta_c = 0$ in the synchronous gauge (by definition), and θ_ν is the transfer function labelled by `t_ncdm[0]` (if there is only one non-cold-dark-matter species).

3 Modifying class

3.1 Implement η_b as a new input parameter

Implement the baryon asymmetry parameter η_b as a new input parameter, as an alternative to ω_b or Ω_b. Note that $\Omega_b h^2 = 1.81 \cdot 10^6 \eta_b \left(\frac{T_{\gamma,0}}{K}\right)^3$. The temperature $T_{\gamma,0}$ in Kelvins is called `pba->T_cmb` in the code. Of course CLASS should accept only one out of the three input parameters (η_b, ω_b, Ω_b), otherwise there would be some redundancy. To check how to code this, you can look at what is done for the photon density input...

3.2 A very simple modification of gravity

There exist several ways to parametrize modifications of gravity. For instance, people often study the effect of a function $\mu(k,\tau)$ inserted in the Poisson equation, giving in the synchronous gauge:

$$k^2\eta - \frac{1}{2}\frac{a'}{a}h' = -\mu(k,\tau)\ 4\pi G a^2 \bar{\rho}_{\rm tot}\delta_{\rm tot}\ .$$

The perturbed Einstein equations are defined in a single place, in `perturb_einstein(...)`. Localize the above equation and implement, for instance, $\mu = 1 + a^3$. Or if you want to do it in a nicer way, implement

$\mu = 1 + \alpha\, a^3$ where α is a new input parameter that should be declared within the perturbation structure, read in `input.c`, and assigned a default value of 0.

Once the modification is done, print the evolution of ϕ and ψ in the standard and modified models (with $\alpha = 1$), and conclude that the C_l^{TT}'s should be affected only through the late ISW effect. Get a confirmation by comparing directly the C_l's of the two cases.

3.3 Comparing the evolution of perturbations in the newtonian and synchronous gauge

We have seen (e.g. through the notebook `one_k`) that it is possible to output the evolution of all perturbations for a given k as a function of conformal time τ.

1. Try to localize the function in which this output is actually defined (i.e., for instance, the place specifying that if we want such an output, the code should store the value of quantities like `delta_g, theta_g, shear_g`, which are the first three multipoles of the photon temperature perturbations). To find the answer without knowing the code by heart, you can follow this typical strategy: logically, to which module should this function belong? In which file `*.c` should it be written? Try to localize the description of this file in the *online automatic documentation*. Browse the detailed description of all the functions in this file. Find the one answering the question.

2. By browsing the doc and/or the file itself, try to understand how gauges are dealt with in this function, in particular: if we run either in synchronous or newtonian gauge, will the code output `delta_g, theta_g, shear_g` in that gauge, or will it systematically convert the output to one of the two gauges?

3. After eventually modifying the code (only a little bit), write a short notebook in order to plot the evolution of `delta_g, theta_g, shear_g` as a function of time for the mode $k = 0.01$ Mpc^{-1} in the two gauges. Check that the behavior of `delta_g` and `theta_g` is very different in these two gauges on super-Hubble scales, while the photon shear `shear_g` is gauge-invariant.