CLASS Exercises

Julien Lesgourgues & Thomas Tram
julien.lesgourgues@cern.ch, thomas.tram@epfl.ch

October 27, 2014

Written solutions will be made available progressively, after each session.

Exercise 0: using the standard output of CLASS

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.

1. Write an 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, and to tune a few other details. This can be done by adding in the input file:

   ```
   N_ur = 2.03351
   N_ncdm = 1
   m_ncdm = 0.06
   T_ncdm = 0.715985            # optional: this is the default since class v2.4.1
   T_cmb = 2.2755               # optional: this is the default since class v2.4.1
   reionisation_width = 0.5     # optional: this is the default since class v2.4.1
   ```

   (you will understand the meaning of the `..._ncdm` parameters in a forthcoming lecture). 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 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. 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` = ...

Exercise 1a: Comparison between lensed and unlensed temperature spectrum

Check the difference between the lensed and unlensed C_l^{TT} of scalars, to see the effect of smoothing of the maxima and minima of the spectrum, and the extra damping induced by lensing on small scales.

Exercise 1b: Comparison between lensed and unlensed BB spectrum

Check the difference between the lensed and unlensed C_l^{BB} in presence of tensor modes, to see that B modes are dominated by lensing on small scales. Use $r = 0.2$ like in BICEP results! To fix this value of the tensor to scalar ratio, just add this line to your input file: `r = 0.2`

Exercise 1c: Comparison between adiabatic and isocurvature CMB spectra

You may switch off the tensors of the previous exercise. Check now the difference between the unlensed C_l^{TT} of scalar modes for adiabatic and CDM isocurvature (CDI) initial conditions (with index $n_{cdi} = 1$), to check that peaks are suppressed in amplitude and shifted in scale. In order to enhance the isocurvature spectrum, you may use the cdi isocurvature fraction `f_cdi = 2`, together with `n_cdi = 1`.

Do the same with NID isocurvature modes (with index $n_{nid} = 1$) to check that the suppression in amplitude is less pronouced and the phase of NID and CDI are different. To enhance the isocurvature spectrum, you may use `f_nid = 4`, together with `n_nid = 1`.

Exercise 1d: Comparison between linear and non-linear matter spectrum

You may switch off the isocurvature modes of the previous exercise. Check now the difference between the linear and non-linear matter power spectrum at $z = 0$ and $z = 2$, to see that at low redshift non-linear corrections are present on larger scales.

Exercise 1e: 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 temparature $T_{\gamma,0}$ is called `pba->T_cmb` in the code.

Exercise 1f: Implement σ_8 as a new input parameter

Implement σ_8 (the amplitude of fluctuations in a sphere of radius 8 Mpc) as a new input parameter, as an alternative to A_s. Currently, σ_8 is computed by the `spectra.c` module, and stored in `psp->sigma8`. Note that this computation takes place only if `output =` includes at least `mPk`, and is accurate enough only if `P_k_max_1/Mpc` is set at least to 1.

Exercise 2a: Printing and plotting background quantities

Reproduce this plot from the Dodelson book on *Modern Cosmology*, using the plotting software of your choice (CPU.py, gnuplot, IDL, matlab, matplotlib functions if you use python and the classy wrapper, etc...). You can use whatever reasonable values of ΛCDM parameters (for instance, $h = 0.7$, $\Omega_b = 0.05$, $\Omega_{cdm} = 0.95$). Note that Dodelson plots the three cosmological distances d_X in units of $[1/H_0]$, which is equivalent to saying that he plots the dimensionless product $d_X H_0$. Since $H_0 = h/3000$ Mpc^{-1}, if you use directly the output of the code in units of Mpc, your y-axis will differ from that of Dodelson by a factor $h/3000$ (e.g. $(0.7/3000)$).

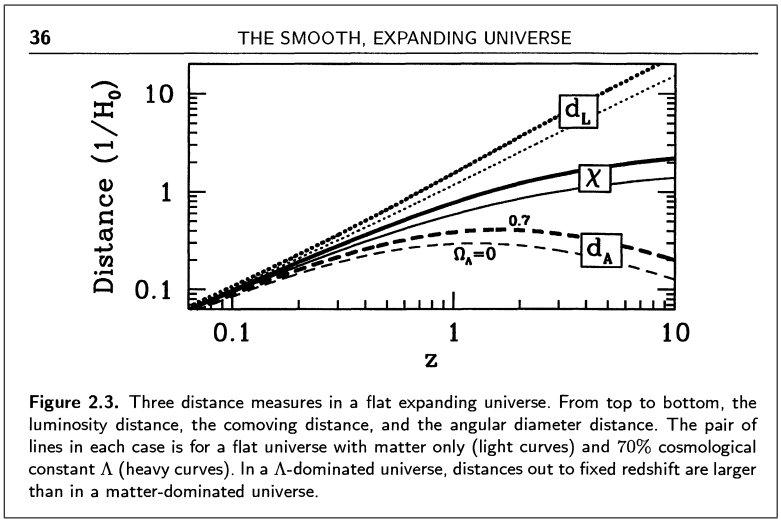

Figure 2.3. Three distance measures in a flat expanding universe. From top to bottom, the luminosity distance, the comoving distance, and the angular diameter distance. The pair of lines in each case is for a flat universe with matter only (light curves) and 70% cosmological constant Λ (heavy curves). In a Λ-dominated universe, distances out to fixed redshift are larger than in a matter-dominated universe.

Exercise 2b: Adding a species in the background module

Introduction. Many types of cosmological species are already available in CLASS, but sometimes it will be necessary to add another species. In this exercise you will add a fluid with equation of state parameter w to CLASS. A general fluid has already been implemented in CLASS with equation of state parameter $w = w_0 + w_a(1 - a/a_0)$ and arbitrary sound speed c_s^2. Nevertheless the exercise is not entirely pointless, because there could be situations where the model contains (or can be modelled by) two uncoupled fluids with different equation of state parameters.

Reading and storing new parameters. We need CLASS to read two additional parameters from the `.ini` file: `Omega_efld` and `w0_efld` (there is no need to call it `w0` because it is constant in time). (`efld` ≡ extra fluid). First add the two new parameters (`Omega0_efld`, `w0_efld`) to the background structure defined in `background.h`. Then open `input.c` and scroll down to `input_read_parameters()` which begins at around line 193. We need to add a couple of new lines to this function, and in principle they could be added almost anywhere inside this function. However, all the species are written in the same order all over the code, so it is nice to add the new species to the "order of species" and then stick

to this convention. The order is {photons, baryons, ultrarelativistic species, cold dark matter, non-cold dark matter, curvature, lambda, fluid}. Since we are adding another fluid, it would fit nicely on either side of the existing fluid. However in this particular function, the extra fluid part must go before the lambda and fluid part, since the values assigned here will depend on `Omega_tot`.

Since everything related to the fluid species has `_fld` at the end, you can just search for `_fld` in your editor. Just before the line

```
/* Omega_0_lambda (cosmological constant), (...) */
```

add a comment about the species you are adding. Now we must read the two input values of (`Omega0_efld`, `w0_efld`): to this end we will utilise the macro[2] `class_read_double(name,destination)`. `name` should be a string, e.g. `"Omega_efld"` and `destination` should then be `pba->Omega0_efld`. We must also add `Omega0_efld` to the total density:

```
Omega_tot += pba->Omega0_efld;
```

We must now set default values for the new parameters. Continue searching for instances of `_fld` until you find

```
pba->Omega0_fld = 0.;
```

Do the same for `pba->Omega0_efld` and put some default value of the equations of state parameter as well.

Modifying the background evolution. Many small changes need to be made to `background.c` and `background.h`. The strategy is to search the file for a similar species, in this case `_fld` and replicate (read: copy-paste-modify) the lines. Note that the evolution of the `_efld` species is uniquely determined by the scale factor, so we do not need to evolve the energy density in time. In CLASS-language, `rho_efld` is known as an {`A`}-variable: it is an analytic function of {`B`}-variables. (In the vanilla vase, the only `B`-variable is the scale factor.) Going through `background.c`, you will realise that you also need to define a number of additional parameters to `background.h`. Be careful to change all instances of `_fld` to `_efld` if you copy-paste!

Compiling and running. Recompile with `make clean` and `make -j class` (the `make clean` is crucial because you have modified at least one `*.h` file; the `-j` speeds up compilation by parallelising it on your computer). You can now create a `.ini` files where one uses e.g. the parameters (`Omega_efld = 0.1`, `w0_efld = 0.1` and `root = efluid_`), to simulate the case $w = 0.1$. Run the code and check that the output `efluid_background.dat` is correctly created, with one columns for the extra fluid.

Checking the result by comparing with the in-built fluid component. You can duplicate the previous `.ini` to a new one, where the same role should be played by the in-built fluid component (`Omega_fld = 0.1`, `w0_fld = 0.1`, `wa_fld = 0.` and `root = fluid_`, while the parameter `Omega_efld` should either be set to zero, or left blank, or commented out).

Check that with this new input file, when you leave the `output` field blank (or you do not pass it at all), the code runs well, while if you pass e.g. `output = tCl [, ...]`, the code complains. This is normal, and a good way to ge familiar with the error management system in CLASS. Have a look at the

[2] A macro in C is a few lines of code which will be pasted directly into the source code by the preprocessor before compilation. The macros for reading parameters are defined in `input.h`.

error message. In which module did the error occur? You can have a look also at the exact line producing the error.

The reason for the error is that for `w0_fld>0`, the code cannot find initial conditions for the fluid perturbations in the standard way: hence the perturbation module contains a protection against this case. Since we are only interested in the background evolution, we can avoid this error by not computing output spectra, i.e. leaving the `output` field blank. (However, after doing so, we can run with values like `w0_fld = 0.1`, but not with much higher values; if you would try e.g. with 0.4, you would get another error message, coming from another protection in the background module).

You should now finally compare the energy densities `rho_fld` and `rho_efld` which should be identical, for the same values of $\Omega_{\rm efld}$ and $w_{\rm efld} = 0.1$. Plot this energy density together with `rho_cdm` and `rho_ur`, as a function of proper time. Your plot should be similar to figure 1, and if the two runs give identical results, in means that the `_efld` has been implemented correctly! After this check, you could in principle run with two different fluids at the same time...

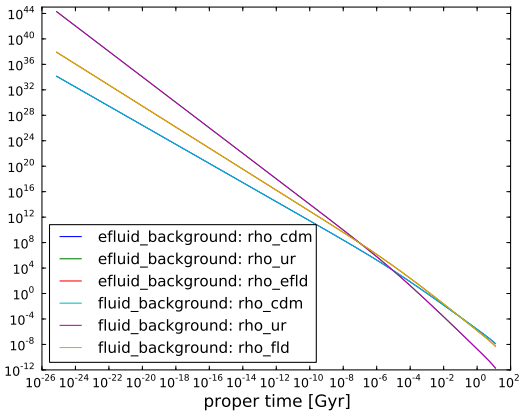

Figure 1: Energy densities of CDM, massless neutrinos and a fluid with equation of state parameter $w = 0.1$.

Exercise 3: A very simple modification of gravity

There exist several ways to parametrise 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(...)`. Localise the above equation and implement, for instance, $\mu = 1 + a^3$. Print the evolution of ϕ and ψ in the standard and modified models, 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 (printed in the files `<root>_cl.dat`).