Solutions to the Homework in Lecture 16
Fundamentals of Observational Cosmology

These notes provide worked solutions to the homework problems in Lecture 16. A compact Julia example for the practical CAMB workflow is provided separately in the file `lecture16_camb_workflow.jl`.

Problem 1. Translating CAMB notation into the CDM continuity equation

Question. Starting from the synchronous-gauge relation $\delta_c' = -h'/2$ and the CAMB definition $z = h'/(2k)$, show explicitly that the code equation `clxcdot = -k*z` is equivalent to the Ma-Bertschinger CDM continuity equation.

Solution. The Ma-Bertschinger cold-dark-matter continuity equation in synchronous gauge is

$$\delta_c' = -\frac{1}{2}h'.$$

The lecture notes identify the CAMB variable `z` as

$$z = \frac{h'}{2k}.$$

Multiply both sides by k:

$$kz = \frac{h'}{2}.$$

Therefore

$$-kz = -\frac{h'}{2}.$$

Since the CAMB variable `clxc` is just the CDM density contrast δ_c, its time derivative `clxcdot` is δ_c'. Hence

$$\texttt{clxcdot} = -k\,\texttt{z}$$

is exactly

$$\delta_c' = -\frac{h'}{2}.$$

So the code line is not a different equation at all; it is simply the synchronous-gauge CDM continuity equation written in CAMB's internal variable names.

Problem 2. Why line-of-sight integration is fast

Question. Explain in words why the line-of-sight method is faster than evolving every multipole to the present time for every ℓ. In your answer, identify which part of the calculation depends mainly on geometry and which part depends on the cosmological model.

Solution. The brute-force approach would evolve the full photon hierarchy,

$$\Theta_0, \Theta_1, \Theta_2, \ldots, \Theta_{\ell_{\max}},$$

all the way to today for every Fourier mode k. That is expensive because the number of coupled equations becomes very large once one wants CMB spectra up to high multipole.

The line-of-sight method reorganizes the problem. Instead of evolving every multipole to the present time, one evolves the physically relevant source functions in time and then projects them onto observed multipoles at the end:

$$\Theta_\ell(k,\tau_0) = \int_0^{\tau_0} \mathrm{d}\tau \, S_T(k,\tau)\, j_\ell\big[k(\tau_0-\tau)\big].$$

This is faster because the difficult cosmological evolution is compressed into the source function $S_T(k,\tau)$, while the angular structure is inserted afterward through the spherical Bessel projection.

The split is:

- **Mainly cosmology-dependent part:** the source function $S_T(k,\tau)$. It depends on the background expansion, recombination history, reionization, perturbation evolution, and all the physical parameters of the cosmological model.
- **Mainly geometry-dependent part:** the projection kernel $j_\ell[k(\tau_0-\tau)]$ and related geometric factors. This part describes how perturbations at comoving distance $\tau_0-\tau$ project onto angles on the sky.

So the line-of-sight method is efficient because it separates "what the universe does" from "how that signal projects onto the sky." Once the sources are known, one can obtain many ℓ values by relatively cheap projection rather than by evolving a huge hierarchy of differential equations for every multipole.

Problem 3. Why tight coupling implies $\theta_b \simeq \theta_\gamma$

Question. In the tight-coupling era, why is it reasonable to set $\theta_b \simeq \theta_\gamma$ to leading order? What physical process enforces this relation?

Solution. Before recombination the universe contains many free electrons, so photons scatter extremely frequently through Thomson scattering. The interaction rate $\dot\kappa$ is large, which means that the photon mean free path is very short.

This appears directly in the Euler equations through drag terms of the form

$$\dot\kappa(\theta_\gamma-\theta_b) \qquad \text{or} \qquad \dot\kappa(\theta_b-\theta_\gamma).$$

If $\dot\kappa$ is very large, any sizable velocity difference between baryons and photons would produce a very large restoring term. The system therefore quickly drives the velocity slip to a tiny value:

$$\theta_b-\theta_\gamma = \mathcal{O}(\dot\kappa^{-1}).$$

To leading order one can therefore set

$$\boxed{\theta_b \simeq \theta_\gamma.}$$

Physically, the process enforcing this relation is **Thomson scattering** of photons on electrons, together with the electromagnetic coupling that keeps electrons and baryons moving together as one tightly coupled fluid. In short: rapid scattering continuously exchanges momentum and locks the baryon and photon velocities together.

Problem 4. A minimal CAMB workflow for C_ℓ^{TT} and $P(k, z = 0)$

Question. Write down a minimal CAMB workflow to compute both C_ℓ^{TT} and the linear matter power spectrum at $z = 0$. You do not need exact code syntax, but you should name the key objects or steps in the right order.

Solution. A minimal workflow is:

1. **Import CAMB** from the frontend you are using.
2. **Create a parameter object**, usually called something like `CAMBparams`.
3. **Set the background cosmology**, for example H_0, $\Omega_b h^2$, $\Omega_c h^2$, neutrino mass, and curvature.
4. **Set the primordial spectrum**, at least A_s and n_s.
5. **Set the CMB calculation range**, for example with a chosen maximum multipole $\ell_{\max}$.
6. **Request matter transfer information** by enabling transfer outputs and specifying a matter-power setup such as `redshifts=[0.0]` and a chosen `kmax`.
7. **Run the solver** to obtain a results object, for example through something like `get_results(pars)`.
8. **Extract CMB power spectra** from the results object, then read off the TT spectrum from the returned array or dictionary.
9. **Extract the linear matter power spectrum** at $z = 0$ from the same results object using a matter-power method or interpolator.

In compact symbolic form, the logic is

$$\text{parameters} \longrightarrow \text{results object} \longrightarrow \begin{cases} C_\ell^{TT}, \\ P(k, z = 0). \end{cases}$$

A more concrete ordering is:

`import camb` → `pars = CAMBparams()` → `set_cosmology()` → `InitPower.set_params()` → `set_for_lmax()` → `WantTransfer = True` → `set_matter_power(redshifts=[0.0], kmax=...)` → `results = get_results(pars)` → `get_cmb_power_spectra()` and `get_matter_power_spectrum()`.

That is the essential modern workflow. The same parameter object controls both the CMB and matter calculations.