Milestone 1

Elisabeth Strøm[1]
AST5220 15.02.2018

[1] Institute of Theoretical Astrophysics, University of Oslo

Abstract. We study the evolution of the Hubble parameter, H, the conformal time, η, and the density parameters Ω of dark matter, baryons, radiation and dark energy, as a function of the scale factor a. We are able to find the correct value of H today, $H_0 = 70.00$ km s^{-1} Mpc^{-1} from our numerical computations, and see also that it was much larger in the past.

1. Introduction

In this project we will numerically decide the time evolution of the Hubble constant and the various constituents of the Universe, meaning the various forms of matter and energy. In other words we wish to look at the evolution of the uniform background of the Universe. This is the first step on the way of deciding the CMB power spectrum. In the next section we will go through the necessary theoretical background, and how we implement this numerically to find the evolution of the desired physical parameters and quantities. Note that we will be using the provided framework code. In the third and final section, we show the results of our calculations.

2. Method

Here we present the methods we have used to obtain the results in Section 3.

2.1. Theory

The Friedmann-Robertson-Walker metric for flat space is defined as

$$\begin{aligned} \mathrm{d}s^2 &= -c^2\mathrm{d}t^2 + a^2(t)(\mathrm{d}r^2 + r^2(\mathrm{d}\theta^2 + sin^2\theta\mathrm{d}\phi^2)) && (1)\\ &= a^2(t)(-\mathrm{d}\eta^2 + \mathrm{d}r^2 + r^2(\mathrm{d}\theta^2 + sin^2\theta\mathrm{d}\phi^2)), && (2)\end{aligned}$$

where $a(t)$ is the scale factor, and η is the conformal time defined as

$$\frac{\mathrm{d}\eta}{\mathrm{d}a} = \frac{c}{a^2H} = \frac{c}{a\mathcal{H}}. \quad (3)$$

where c is the speed of light in vacuum, H is the Hubble parameter, and where we have defined the scaled Hubble parameter $\mathcal{H} \equiv aH$. This is the background cosmology for this project.

From the first Friedmann equation, we get an expression for the Hubble parameter,

$$H = H_0\sqrt{(\Omega_{\mathrm{b},0} + \Omega_{\mathrm{m},0})a^{-3} + (\Omega_{\mathrm{r},0} + \Omega_{\nu,0})a^{-4} + \Omega_{\Lambda,0}}, \quad (4)$$

and the scaled Hubble parameter,

$$\mathcal{H} = H_0\sqrt{(\Omega_{\mathrm{b},0} + \Omega_{\mathrm{m},0})a^{-1} + (\Omega_{\mathrm{r},0} + \Omega_{\nu,0})a^{-2} + \Omega_{\Lambda,0}a^2}, \quad (5)$$

where $H_0 = 70$ km s^{-1} Mpc^{-}1 is the Hubble parameter of today, $\Omega_\mathrm{b} = 0.046$, $\Omega_\mathrm{m} = 0.224$, $\Omega_\mathrm{r} = 8.3 \cdot 10^{-5}$, $\Omega_\nu = 0$ and $\Omega_\Lambda = 0.730$ are the density parameters of today of baryonic matter, dark matter, radiation, neutrinos, and dark energy, respectively. We will do our calculations without neutrinos. The subscript 0 means the parameter's value at present day.

These density parameters evolve in time as follows;

$$\Omega_\mathrm{b} = \Omega_{\mathrm{b},0}\left(\frac{H}{H_0}\right)^2 a^{-3}, \quad (6)$$

$$\Omega_\mathrm{m} = \Omega_{\mathrm{m},0}\left(\frac{H}{H_0}\right)^2 a^{-3}, \quad (7)$$

$$\Omega_\mathrm{r} = \Omega_{\mathrm{r},0}\left(\frac{H}{H_0}\right)^2 a^{-4}, \quad (8)$$

$$\Omega_\Lambda = \Omega_{\Lambda,0}\left(\frac{H}{H_0}\right)^2. \quad (9)$$

For convenience, we will not be working with the scale factor a, but its natural logarithm,

$$x = \ln a. \quad (10)$$

We define $a_0 = 1$ to be the scale factor of present day, meaning that $x \leq 0$. Equation 3 can then be rewritten as

$$\frac{\mathrm{d}\eta}{\mathrm{d}x} = \frac{c}{\mathcal{H}} \quad (11)$$

We will solve this ordinary differential equation (ODE) numerically.

We wish to know how the Hubble parameter H evolve with the scale factor x, and the cosmological redshift z. The cosmological redshift is defined as

$$1 + z = \frac{a_0}{a} = e^{-x}. \quad (12)$$

2.2. Implementation & Algorithm

The objective in this project is to find how the density parameters Ω_X evolve with x, and how H evolve with x and z. To do this, we will have to solve the ODE in Equation 11, the first Friedmann equation 4 and 5, and the equations 6, 7, 8, and 9, which we will do numerically.

We set up a gird of x values which we will use for our computations. The initial and final values of x corresponds to

$a_{\text{initial}} = 10^{-10}$ and $a_{\text{final}} = 1$. The algorithm is

$$x_{i+1} = x_{\text{initial}} + i \cdot \frac{(x_{\text{final}} - x_{\text{initial}})}{n-1}, \tag{13}$$

where we have n gird points. These grid points are denoted `x_eta` in our code.

To find $\eta(x)$ we use a pre-existing ODE solver. To use this method, we need the initial value of $\eta(x)$. We can use that in the early days of the universe, around a_{init}, the cosmos consisted almost entirely of radiation, meaning $\Omega_{\text{r}} = 1$. If we integrate Equation 3, we get

$$\eta_{\text{init}} = \int_0^{a_{\text{init}}} \frac{c}{H_0\sqrt{\Omega_{\text{r}}}} \text{d}a = \frac{a_{\text{init}} c}{H_0}. \tag{14}$$

We also make functions that compute H, $\mathcal{H}$, and $\frac{\text{d}\mathcal{H}}{\text{d}x}$, whereas

$$\frac{\text{d}\mathcal{H}}{\text{d}x} = -\frac{H_0}{H}\left(\frac{1}{2}(\Omega_{\text{b}} - \Omega_{\text{m}})a^{-2} + (\Omega_{\text{r}} + \Omega_{\nu})a^{-3} - \Omega_{\Lambda} a\right). \tag{15}$$

In the ODE solver, we use that the interval between two values is

$$dx = \frac{(x_{\text{initial}} - x_{\text{final}})}{100(n-1)}. \tag{16}$$

2.2.1. Future projects

In future projects we will need a different grid. We set up a grid of x and a values during ($z = 1630 - 614$) and after ($z = 614 - 0$) recombination. In two separate loops with respectively $n_1 - 1$ and n_2 number of iterations , where n_1 and n_2 are the number of grid points during and after recombination ,we use the following algorithm:

$$x_{i+1} = x_{\text{start rec}} + i \cdot \frac{(x_{\text{end rec}} - x_{\text{start rec}})}{n_1 - 1}, \tag{17}$$

$$x_{n_1+i} = x_{\text{end rec}} + i \cdot \frac{(x_0 - x_{\text{end rec}})}{n_2 - 1}, \tag{18}$$

where $x_{\text{start rec}}$, $x_{\text{end rec}}$, and x_0 denote the value of x at the three stated redshifts, $z = 1630$, 614 and 0, respectively. Using the relation in Equation 10, we can make a similar grid of a values. These grid points are denoted `x_t` and `a_t`.

We also spline the η array resulting from solving the ODE, and integrate these for the new x - grid.

The programming code can be found here and in the appendix.

3. Results

The different density parameters are shown in Figure 1 against x. It shows that in the early universe, up until $x = -7.906$, there where radiation domination. This value of x correspond to a redshift of $z \approx 2712$. As Ω_{r} decreases toward 0, Ω_{m} increases, having a maximum value of $\Omega_{\text{m}} \approx 0.825$. We can also see that the dominating matter is mostly non-baryonic, dark matter, as Ω_{b} peaks at just under $\Omega_b = 0.2$ in the same x-interval. Dark matter dominates from $x = -7.906$ to $x = -0.415$, corresponding with a redshift of $z \approx 2712 - 0.660$.

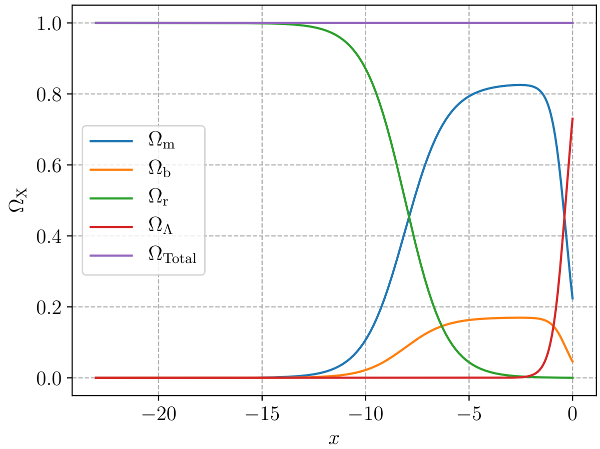

Fig. 1: The different density parameters, Ω, as a function of the natural logarithm of the scale factor, $x = \ln a$. We see that their sum is equal to 1 at all times, which is expected. The graph shows that the Universe was radiation dominated at early times, and that radiation-matter equality occurred at around $x = -7.906$. The radiation density parameter decreases quickly, and is close to zero around $x = -5$. Matter, dark matter, then dominated until matter-dark energy equality at around $x = -0.415$

Dark matter- dark energy equality occurs at $z = 0.660$, which is when Ω_{Λ} becomes larger than Ω_{m}, meaning that in our time at $x = 0$ and $a = 1$, dark energy is dominating the Universe.

In Figure 2, we see that the Hubble parameter was much larger in the past than it is at present. In fact, it seems that it undergoes quite little change in our own time, judging by the flatness of curve at small z and high x. We find that the Hubble parameter value today is $H_0 = 70.00$ km s^{-1} Mpc^{-1}, which is what we would expect.

The conformal time $\eta(x)$ can be seen in Figure 3. The conformal time is a measure of the distance to the particle horizon for a given value of x. We see that in the early days of the universe $\eta(x)$ increased rapidly, although it maintained a very small value. Today, the radius of the observable universe is around 14.3 Gpc (Bars & Terning 2010), while from our graph we get that the conformal time at $x = 0$ is $\eta(0) = 14.5$ Gpc. So our result is in good accordance with previously obtained results.

4. Conclusions

We have numerically calculated how the various density parameters Ω have evolved through time up to present day. The Universe was dominated by radiation up until redshift $z = 2712$, when matter-radiation equality occurred, and dark matter goes on to dominate. The baryonic density parameter Ω_{b} during this time is around 0.2, as opposed to the dark matter density $\Omega_{\text{m}} \approx 0.8$, while $\Omega_{\text{r}} \to 0$. At $z = 0.66$ up until now, dark energy is dominating.

From calculating the evolution of the Hubble parameter, we arrive correctly at the present day value of $H_0 = 70.00$ km s^{-1} Mpc^{-1}, we also find that H was larger in the past.

The conformal time In this project we will numerically decide the time evolution of the Hubble constant and the various constituents of the Universe, meaning the various forms of mat-

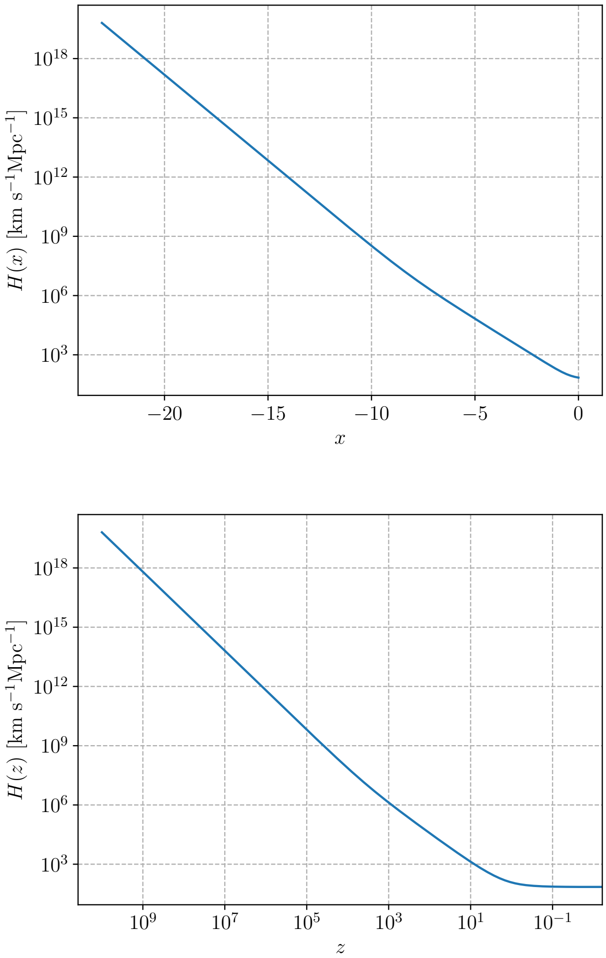

Fig. 2: The Hubble parameter shown in log-scale against the natural logarithm of the scale factor, $x = \ln a$ (top figure), and against the redshift, z (lower figure). Note the flipped x-axis in the redshift graph, and the log-scale on the y-axis. We see that the Hubble parameter decreases as the scale factor of the universe increases.

ter and energy. In other words we wish to look at the evolution of the uniform background of the Universe. In the next section we will go through the necessary theoretical background, and how we implement this numerically to find the evolution of the desired physical parameters and quantities. Note that we will be using the provided framework code. In the third and final section, we show the results of our calculations.

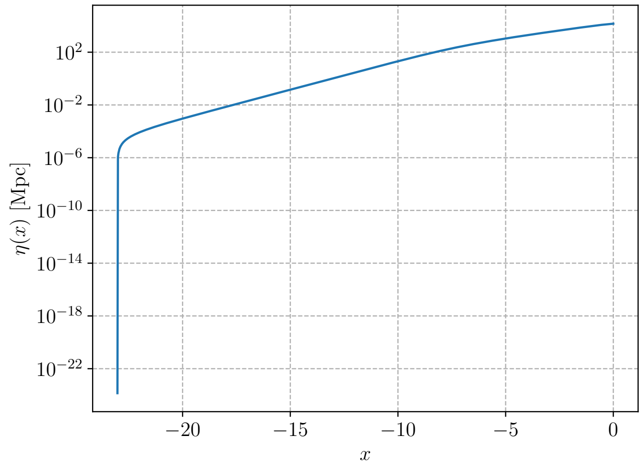

Fig. 3: The conformal time is shown in log-scale on the y-axis against $x = \ln a$ on the x-axis

References

Bars, I. & Terning, J. 2010, Extra dimensions in space and time (New York, NY: Springer)

5. Appendix

Source code

Listing 1: C:/Users/elini/Documents/AST5220/Ast5220/src/time_mod.f90

```
module time_mod
  use healpix_types
  use params
  use ode_solver
  use spline_1D_mod
  implicit none

  integer(i4b)                           :: n_t              ! Number of x-values
  real(dp),  allocatable, dimension(:) :: x_t              ! Grid of relevant x-values
  real(dp),  allocatable, dimension(:) :: a_t              ! Grid of relevant a-values

  integer(i4b)                           :: n_eta            ! Number of eta grid poins
  real(dp),  allocatable, dimension(:) :: x_eta            ! Grid points for eta
  real(dp),  allocatable, dimension(:) :: eta, eta2       ! Eta and eta'' at each grid point

contains

  subroutine initialize_time_mod
    implicit none

    integer(i4b) :: i, n, n1, n2
    real(dp)    :: z_start_rec, z_end_rec, z_0, x_start_rec, x_end_rec, x_0, dx, x_eta1, x_eta2, a_init
    real(dp)    :: x_init, x_int1, x_int2, x_eta_int, eps, step, stepmin, eta_init, rho_c
    real(dP)    :: H_scale, Omega_mx, Omega_bx, Omega_rx, Omega_lambdax, z

    ! Define two epochs, 1) during and 2) after recombination.
    n1          = 200                     ! Number of grid points during recombination
    n2          = 300                     ! Number of grid points after recombination
    n_t         = n1 + n2                 ! Total number of grid points
    z_start_rec = 1630.4d0                ! Redshift of start of recombination
    z_end_rec   = 614.2d0                 ! Redshift of end of recombination
    z_0         = 0.d0                    ! Redshift today
    x_start_rec = -log(1.d0 + z_start_rec) ! x of start of recombination
    x_end_rec   = -log(1.d0 + z_end_rec) ! x of end of recombination
    x_0         = 0.d0                    ! x today

    n_eta       = 1000                    ! Number of eta grid points (for spline)
    a_init      = 1.d-10                  ! Start value of a for eta evaluation
    x_eta1      = log(a_init)             ! Start value of x for eta evaluation
    x_eta2      = 0.d0                    ! End value of x for eta evaluation

    eps         = 1.d-8                      ! spline error limit
    eta_init    = c*a_init/H0 ! eta initial value at a=0

    ! Task: Fill in x and a grids
    allocate(x_t(n_t))
    allocate(a_t(n_t))

    ! x_init = x_start_rec
    x_int1 = (x_end_rec - x_start_rec) /(n1 - 1)
    x_int2 = (x_0 - x_end_rec) /(n2 - 1)

    ! x grid during recombination
    x_t(1) = x_start_rec
    do i=1,n1-1
       x_t(i+1) = x_start_rec + i*x_int1
    end do

    ! x grid after recombination
    do i=1,n2
       x_t(n1+i) = x_end_rec + i*x_int2
    end do

    ! a grid values
```

```fortran
  a_t = exp(x_t) ! x = ln a


  ! Task: 1) Compute the conformal time at each eta time step
  !      2) Spline the resulting function, using the provided "spline" routine in spline_1D_mod.f90
  allocate(x_eta(n_eta))
  allocate(eta(n_eta))
  allocate(eta2(n_eta))

  ! x_eta grid
  x_eta_int = (x_eta2 - x_eta1)/(n_eta - 1)
  x_eta(1) = x_eta1
  do i=1,n_eta-1
     x_eta(i+1) = x_eta1 + i*x_eta_int
  end do

  ! integrating to find eta

  step = abs((x_eta(1) - x_eta(2))/ 100.d0)
  stepmin = abs((x_eta(1) - x_eta(2))/ 10000.d0)

  eta(1) = eta_init
  do i=1,n_eta-1
     eta(i+1) = eta(i)
     call odeint(eta(i+1:i+1), x_eta(i), x_eta(i+1), eps, step, stepmin, derivs, bsstep, output)
  end do

  ! calling spline on eta
  call spline(x_eta, eta, 1d30, 1d30, eta2)



  ! write stuff to file - x_eta. eta. eta splint, H, Omegas
  open (unit=1, file = 'xt_eta_t.dat', status='replace')
  open (unit=2, file = 'omega_mbrl.dat', status='replace')
  open (unit=3, file = 'xeta_eta.dat', status='replace')
  open (unit=4, file = 'xeta_z_H.dat', status='replace')

  do i=1,n_t
     write (1,'(2(E17.8))') x_t(i), get_eta(x_t(i))
  end do

  do i=1, n_eta

    ! calculate and write Omegas
    H_scale      = H_0/get_H(x_eta(i))
    Omega_mx     = Omega_m     * H_scale**2 * exp(x_eta(i))**(-3)
    Omega_bx     = Omega_b     * H_scale**2 * exp(x_eta(i))**(-3)
    Omega_rx     = Omega_r     * H_scale**2 * exp(x_eta(i))**(-4)
    Omega_lambdax = Omega_lambda * H_scale**2

    z = exp(-x_eta(i))-1
    !write (*,'(3(E17.8))') x_eta(i), z, get_H(x_eta(i)),z
    write (2,'(4(E17.8))') Omega_mx, Omega_bx, Omega_rx, Omega_lambdax
    write (3,'(2(E17.8))') x_eta(i), eta(i)
    write (4,'(3(E17.8))') x_eta(i), z, get_H(x_eta(i))

  end do

  do i=1,4 ! close files
    close(i)
  end do

end subroutine initialize_time_mod


subroutine derivs(x, eta, detadx)
  ! we define d eta/d x
  use healpix_types
  implicit none
```

```fortran
    real(dp),              intent(in) :: x
    real(dp), dimension(:), intent(in) :: eta
    real(dp), dimension(:), intent(out) :: detadx

    detadx = c/get_H_p(x)
  end subroutine derivs

  subroutine output(x, y)
    use healpix_types
    implicit none
    real(dp),              intent(in) :: x
    real(dp), dimension(:), intent(in) :: y
  end subroutine output

  ! Task: Write a function that computes H at given x
  function get_H(x)
    implicit none

    real(dp), intent(in) :: x
    real(dp)             :: get_H
    real(dp)             :: a
    a = exp(x)
    get_H = H_0*sqrt((Omega_b+Omega_m)*a**(-3) + (Omega_r + Omega_nu)*a**(-4) + Omega_lambda)
  end function get_H

  ! Task: Write a function that computes H' = a*H at given x
  function get_H_p(x)
    implicit none

    real(dp), intent(in) :: x
    real(dp)             :: get_H_p
    real(dp)             :: a
    a = exp(x)
    get_H_p = a*H_0*sqrt((Omega_b+Omega_m)*a**(-3) + (Omega_r + Omega_nu)*a**(-4) + Omega_lambda)

  end function get_H_p

  ! Task: Write a function that computes dH'/dx at given x
  function get_dH_p(x)
    implicit none

    real(dp), intent(in) :: x
    real(dp)             :: get_dH_p
    real(dp)             :: a
    a = exp(x)
    get_dH_p = -H_0*(0.5d0*(Omega_b-Omega_m)*a**(-2) + (Omega_r + Omega_nu)*a**(-3) -
        Omega_lambda*a)/sqrt((Omega_b+Omega_m)*a**(-3) + (Omega_r + Omega_nu)*a**(-4) + Omega_lambda)
  end function get_dH_p

  ! Task: Write a function that computes eta(x), using the previously precomputed splined function
  function get_eta(x_in)
    implicit none

    real(dp), intent(in) :: x_in
    real(dp)             :: get_eta
    get_eta = splint(x_eta, eta, eta2, x_in)

  end function get_eta

end module time_mod
```