AST 3220, project 1, spring 2023

Problem 1

CE for quintessence

$$\dot{\rho}_\phi = -3H(\rho_\phi + p_\phi) = -3H\left(\rho_\phi + \frac{p_\phi}{\rho_\phi}\rho_\phi\right)$$
$$= -3H(1+\omega_\phi)\rho_\phi$$

This is a separable differential equation:

$$\frac{d\rho_\phi}{dt} = -3H(1+\omega_\phi)\rho_\phi = -\frac{3}{a}\frac{da}{dt}(1+\omega_\phi)\rho_\phi$$

$$\Rightarrow \frac{d\rho_\phi}{\rho_\phi} = -3(1+\omega_\phi)\frac{da}{a}$$

$$\Rightarrow \int_{\rho_{\phi 0}}^{\rho_\phi} \frac{d\rho_\phi'}{\rho_\phi'} = -3\int_{a_0}^{a}(1+\omega_\phi)\frac{da'}{a'}$$

Substitution:

$$1+z' = \frac{a_0}{a'} \Rightarrow a' = \frac{a_0}{1+z'}$$

$$a' = a \Rightarrow 1+z' = \frac{a_0}{a} = 1+z$$

$$a' = a_0 \Rightarrow 1+z' = 1 \Rightarrow z' = 0$$

$$da' = -\frac{a_0}{(1+z')^2}dz'$$

Then

$$\ln\left(\frac{\rho_\phi}{\rho_{\phi 0}}\right) = -3\int_0^z [1+\omega_\phi(z')]\frac{(1+z')}{a_0}\left[-\frac{a_0}{(1+z')^2}dz'\right]$$

$$= 3\int_0^z \frac{1+\omega_\phi(z')}{1+z'}dz'$$

and the result in the text follows.

Problem 2 We have

$$\rho_\phi = \frac{1}{2}\dot{\phi}^2 + V(\phi)$$

$$p_\phi = \frac{1}{2}\dot{\phi}^2 - V(\phi)$$

Then $$\rho_\phi + p_\phi = \dot{\phi}^2$$

and $$\dot{\rho}_\phi = \frac{d}{dt}\left[\frac{1}{2}\dot{\phi}^2 + V(\phi)\right]$$

$$= \frac{1}{2}\cdot 2\dot{\phi}\ddot{\phi} + \frac{dV}{d\phi}\cdot\frac{d\phi}{dt}$$

$$= \dot{\phi}\ddot{\phi} + \dot{\phi}V'(\phi)$$

Insert in the CE:

$$\dot{\phi}\ddot{\phi} + \dot{\phi}V'(\phi) = -3H\dot{\phi}^2$$

and after dividing by $\dot{\phi}$ *) and rearranging the terms,

$$\ddot{\phi} + 3H\dot{\phi} + V'(\phi) = 0$$

*) We have to assume $\dot{\phi} \neq 0$ for do that.
But if $\dot{\phi} = 0$, ϕ = constant,

$$\rho_\phi = V(\phi) = \text{constant},$$

and $$p_\phi = -V(\phi) = -\rho_\phi$$

so the scalar field is then equivalent to a cosmological constant, which we are not interested in.

Problem 3

FI: $(K=0)$

$$\left(\frac{\dot a}{a}\right)^2 = \frac{8\pi G}{3}\left(\rho_m+\rho_r+\rho_\phi\right)$$

$$= \frac{\kappa^2}{3}\left(\rho_m+\rho_r+\frac{1}{2}\dot\phi^2+V(\phi)\right)$$

FII: ρ_m^{0}, $\rho_r^{w_r\rho_r}$

$$\frac{\ddot a}{a} = -\frac{4\pi G}{3}\left[\rho_m+\rho_r+\rho_\phi+3p_m+3p_r+3p_\phi\right]$$

$$= -\frac{\kappa^2}{6}\left[\rho_m+\rho_r(1+3w_r)+\frac{1}{2}\dot\phi^2+V(\phi)+\frac{3}{2}\dot\phi^2-3V(\phi)\right]$$

$$= -\frac{\kappa^2}{6}\left[\rho_m+\rho_r(1+3w_r)+2\dot\phi^2-2V(\phi)\right]$$

Since $H=\frac{\dot a}{a}$,

$$\dot H = \frac{d}{dt}\left(\frac{\dot a}{a}\right) = \frac{\ddot a a-\dot a^2}{a^2} = \frac{\ddot a}{a}-\left(\frac{\dot a}{a}\right)^2$$

$$= \cancel{\frac{\kappa^2}{3}\left[\rho_m+\rho_r+\frac{1}{2}\dot\phi^2+V(\phi)\right]}$$

$$= -\frac{\kappa^2}{6}\left[\rho_m+\rho_r(1+3w_r)+2\dot\phi^2-2V(\phi)\right]$$

$$-\frac{\kappa^2}{3}\left[\rho_m+\rho_r+\frac{1}{2}\dot\phi^2+V(\phi)\right]$$

$$= -\frac{\kappa^2}{6}\left[\rho_m+\rho_r+3w_r\rho_r+2\dot\phi^2-2V(\phi)\right.$$

$$\left.+2\rho_m+2\rho_r+\dot\phi^2+2V(\phi)\right]$$

$$= -\frac{\kappa^2}{6}\left[3\rho_m+3\rho_r+3w_r\rho_r+3\dot\phi^2\right]$$

$$= -\frac{\kappa^2}{2}\left[\rho_m+\rho_r(1+w_r)+\dot\phi^2\right]$$

Problem 4

We have

$$\Omega_i = \frac{\rho_i}{\rho_c} = \frac{\rho_i}{\frac{3H^2}{k^2}} = \frac{k^2 \rho_i}{3H^2}$$

Using the definition of the x_i's:

$$x_1^2 + x_2^2 = \frac{k^2 \dot{\phi}^2}{6H^2} + \frac{k^2 V}{3H^2}$$

$$= \frac{k^2}{3H^2}\left(\frac{1}{2}\dot{\phi}^2 + V\right) = \frac{k^2 \rho_\phi}{3H^2}$$

$$= \underline{\underline{\Omega_\phi}}$$

$$x_3^2 = \frac{k^2 \rho_r}{3H^2} = \underline{\underline{\Omega_r}}$$

From FT with $k = 0$

$$H^2 = \frac{k^2}{3}(\rho_m + \rho_r + \rho_\phi)$$

$$\Rightarrow \quad 1 = \frac{k^2}{3H^2}\rho_m + \frac{k^2}{3H^2}\rho_r + \frac{k^2}{3H^2}\rho_\phi$$

$$= \Omega_m + x_3^2 + x_1^2 + x_2^2$$

$$\Rightarrow \quad \underline{\Omega_m = 1 - x_1^2 - x_2^2 - x_3^2}$$

Problem 5

Equation (8):

$$\dot{H} = -\frac{k^2}{2}\left[\rho_m + \rho_r(1+w_r) + \dot{\phi}^2\right]$$

$w_r = \frac{1}{3}$, so

$$\dot{H} = -\frac{k^2}{2}\left[\rho_m + \frac{4}{3}\rho_r + \dot{\phi}^2\right]$$

$$\Rightarrow \quad \frac{\dot{H}}{H^2} = -\frac{k^2}{2H^2}\left(\rho_m + \frac{4}{3}\rho_r + \dot{\phi}^2\right)$$

$$= -\frac{1}{2}\cdot 3\,\frac{k^2}{3H^2}\left(\rho_m + \frac{4}{3}\rho_r + \dot{\phi}^2\right)$$

$$= -\frac{3}{2}\left(\frac{k^2}{3H^2}\rho_m + \frac{4}{3}\cdot\frac{k^2}{3H^2}\rho_r + \frac{k^2}{3H^2}\dot{\phi}^2\right)$$

$$= -\frac{3}{2}\left(\Omega_m + \frac{4}{3}x_3^2 + 2x_1^2\right)$$

$$= -\frac{3}{2}\left(1 - x_1^2 - x_2^2 - x_3^2 + \frac{4}{3}x_3^2 + 2x_1^2\right)$$

$$= -\frac{3}{2}\left(1 + x_1^2 - x_2^2 + \frac{1}{3}x_3^2\right)$$

$$= -\frac{1}{2}\left(3 + 3x_1^2 - 3x_2^2 + x_3^2\right)$$

Problem 6

$$x_1 = \frac{\kappa \dot\phi}{\sqrt{6} H}$$

and

$$\frac{d}{dN} = \frac{1}{H}\frac{d}{dt}, \quad \text{so}$$

$$\frac{dx_1}{dN} = \frac{1}{H}\frac{d}{dt}\left(\frac{\kappa\dot\phi}{\sqrt{6}H}\right) = \frac{\kappa}{\sqrt{6}}\frac{1}{H}\frac{d}{dt}\left(\frac{\dot\phi}{H}\right)$$

$$= \frac{\kappa}{\sqrt{6}}\frac{1}{H^2}\ddot\phi + \frac{\kappa}{\sqrt{6}}\frac{\dot\phi}{H}\frac{d}{dt}\left(\frac{1}{H}\right).$$

$$= \frac{\kappa}{\sqrt{6}H^2}\left(-3H\dot\phi - V'(\phi)\right) - \underbrace{\frac{\kappa}{\sqrt{6}}\frac{\dot\phi}{H}}_{x_1}\frac{\dot H}{H^2}$$

($-3H\dot\phi$: from eq. (6); $\frac{\dot H}{H^2}$: eq. (16))

$$= -3\underbrace{\frac{\kappa\dot\phi}{\sqrt{6}H}}_{=x_1} - \frac{\kappa V'}{\sqrt{6}H^2} - x_1\left(-\frac{1}{2}\right)\left(3 + 3x_1^2 - 3x_2^2 + x_3^2\right)$$

$$= -3x_1 - \frac{\kappa V'}{\sqrt{6}H^2} + \frac{1}{2}x_1\left(3 + 3x_1^2 - 3x_2^2 + x_3^2\right)$$

We also have

$$\frac{\sqrt{6}}{2}\lambda x_2^2 = \frac{\sqrt{6}}{2}\left(-\frac{V'}{\kappa V}\right)\frac{\kappa^2 V}{3H^2}$$

$$= -\frac{\sqrt{6}}{6}\frac{\kappa V'}{H^2} = -\frac{1}{\sqrt{6}}\frac{\kappa V'}{H^2},$$

So

$$\frac{dx_1}{dN} = -3x_1 + \frac{\sqrt{6}}{2}\lambda x_2^2 + \frac{1}{2}x_1\left(3 + 3x_1^2 - 3x_2^2 + x_3^2\right)$$

Next one:

$$\frac{dx_2}{dN} = \frac{1}{H}\frac{d}{dt}\left(\frac{\kappa\sqrt{V}'}{\sqrt{3}H}\right)$$

$$= \frac{\kappa}{\sqrt{3}}\frac{1}{H}\frac{d}{dt}\left(\frac{\sqrt{V}'}{H}\right)$$

$$= \frac{\kappa}{\sqrt{3}H^2}\frac{d}{dt}\sqrt{V}' + \underbrace{\frac{\kappa}{\sqrt{3}H}\sqrt{V}}_{=x_2}\frac{d}{dt}\left(\frac{1}{H}\right)$$

$$= \frac{\kappa}{\sqrt{3}H^2}\frac{1}{2\sqrt{V}}\frac{dV}{d\phi}\dot{\phi} + x_2\cdot\left(-\frac{\dot{H}}{H^2}\right)$$

$$= \frac{\kappa V'\dot{\phi}}{2\sqrt{3}H^2\sqrt{V}} + \frac{1}{2}x_2\left(3+3x_1^2-3x_2^2+x_3^2\right)$$

Checking that

$$\frac{\kappa V'\dot{\phi}}{2\sqrt{3}H^2\sqrt{V}} = -\frac{\sqrt{6}}{2}\lambda x_1 x_2 :$$

$$-\frac{\sqrt{6}}{2}\lambda x_1 x_2 = -\frac{\sqrt{6}}{2}\left(-\frac{V'}{\kappa V}\right)\frac{\kappa\dot{\phi}}{\sqrt{6}H}\frac{\kappa\sqrt{V}}{\sqrt{3}H}$$

$$= \frac{1}{2\sqrt{3}}\frac{\kappa^2 V'\sqrt{V}\dot{\phi}}{\kappa V H^2}$$

$$= \frac{\kappa V'\dot{\phi}}{2\sqrt{3}H^2\sqrt{V}} \quad /$$

So

$$\frac{dx_2}{dN} = -\frac{\sqrt{6}}{2}\lambda x_1 x_2 + \frac{1}{2}x_2\left(3+3x_1^2-3x_2^2+x_3^2\right)$$

Finally:

$$\frac{dx_3}{dN} = \frac{1}{H}\frac{d}{dt}\left(\frac{k\sqrt{\rho_r}}{\sqrt{3}H}\right)$$

$$= \frac{k}{\sqrt{3}}\frac{1}{H}\frac{d}{dt}\frac{\sqrt{\rho_r}}{H} = \frac{k}{\sqrt{3}H^2}\frac{1}{2\sqrt{\rho_r}}\dot{\rho}_r + \underbrace{\frac{k\sqrt{\rho_r}}{\sqrt{3}H}}_{=x_3}\frac{d}{dt}\frac{1}{H}$$

From the CE for radiation:

$$\dot{\rho}_r = -3H(\rho_r + p_r)$$

$$= -3H\left(1+\omega_r^{1/3}\right)\rho_r = -4H\rho_r$$

So

$$\frac{dx_3}{dN} = \frac{k}{\sqrt{3}H^2}\frac{1}{2\sqrt{\rho_r}}(-4H\rho_r) + x_3\left(-\frac{\dot{H}}{H^2}\right)$$

$$= -2\underbrace{\frac{k\sqrt{\rho_r}}{\sqrt{3}H}}_{=x_3} + \frac{1}{2}x_3\left(3+3x_1^2-3x_2^2+x_3^2\right)$$

$$= -2x_3 + \frac{1}{2}x_3\left(3+3x_1^2-3x_2^2+x_3^2\right)$$

Problem 7

λ = constant

$$\Rightarrow \quad -\frac{V'}{\kappa V} = \lambda$$

$$\frac{dV}{d\phi} = -\lambda\kappa V$$

$$\int_{V_0}^{V(\phi)} \frac{dV}{V} = -\lambda\kappa \int_0^{\phi} d\phi'$$

$$\ln\left[\frac{V(\phi)}{V_0}\right] = -\lambda\kappa\phi$$

$$\Rightarrow \quad \underline{V(\phi) = V_0 e^{-\lambda\kappa\phi}} \quad \text{: exponential potential}$$

For this potential

$$V' = -\lambda\kappa V_0 e^{-\lambda\kappa\phi} = -\lambda\kappa V$$

$$V'' = -\lambda\kappa V' = \lambda^2\kappa^2 V$$

so

$$\Gamma = \frac{VV''}{(V')^2} = \frac{V\cdot\lambda^2\kappa^2 V}{\lambda^2\kappa^2 V^2} = \underline{\underline{1}}$$

Problem 8

$$\frac{d\lambda}{dN} = \frac{1}{H}\frac{d}{dt}\left(-\frac{V'}{\sqrt{2}V}\right)$$

$$= -\frac{1}{\sqrt{2}H}\frac{d}{d\phi}\left(\frac{V'}{V}\right)\frac{d\phi}{dt}$$

$$= -\frac{\dot{\phi}}{\sqrt{2}H}\frac{V''V - (V')^2}{V^2}$$

$$= -\frac{\dot{\phi}}{\sqrt{2}H}\frac{(V')^2}{V^2}\left[\frac{V''V}{(V')^2} - 1\right]$$

$$= -\frac{\sqrt{6}H}{\sqrt{2}}x_1\frac{1}{\sqrt{2}H}\left(\frac{V'}{V}\right)^2(\Gamma - 1)$$

$$\dot{\phi} = \frac{\sqrt{6}H}{\sqrt{2}}x_1 \qquad \left(\frac{V'}{V} = -\lambda\right)$$

$$= -\sqrt{6}x_1\left(\frac{V'}{\sqrt{2}V}\right)^2(\Gamma - 1)$$

$$= -\sqrt{6}\lambda^2(\Gamma - 1)x_1$$

Problem 9

My codes are of limited value for you because I am the only FORTRAN 77-programmer still alive, but they are included at the end of this document to please those of you with an interest in medieval history

In my plots I have $-\ln(1+z)$ along the axis, but this is the same as N:

$$N = \ln\left(\frac{a}{a_0}\right) = \ln\left(\frac{1}{1+z}\right) = -\ln(1+z)$$

N is zero today, and negative at earlier times.

While the densities evolve very similarly in the two models, the EoS parameter does not.

Problem 10

When solving the equations numerically, we find $x_i(N)$, so we get $w_\phi(N)$ straightforwardly.

Since $N = -\ln(1+z)$,

$$dN = -\frac{1}{1+z}dz$$

$$\Rightarrow \frac{dz}{1+z} = -dN$$

So

$$\int_0^z dz' \frac{3[1+w_\phi(z')]}{1+z'}$$

$$= -\int_0^{-\ln(1+z)} 3[1+w_\phi(N)]dN$$

$$= 3\int_{-\ln(1+z)}^{0} [1+w_\phi(N)]dN$$

I use Gauss-Legendre integration, and I interpolate in the numerical solution to find w_ϕ on the integration grid rather than solving the differential equations again.

The Hubble parameter looks very similar in the two models (see figure 5), and they are markedly different from ΛCDM at early times.

Closer to our time they all look the same, and we should expect that since the models need to fit the luminosity distance data.

Problem 11 Equation (3.61) in the lecture notes:

$$t_0 = \int_0^\infty \frac{dz}{(1+z)\,H(z)}$$

$$= -\int_0^{-\infty} \frac{dN}{H(N)} = \int_{-\infty}^{0} \frac{dN}{H(N)}$$

For the power-law potential:

$$H_0 t_0 = 0,99$$

Exponential:

$$H_0 t_0 = 0,97$$

ΛCDM:

$$H_0 t_0 = 0,96$$

∴ Power-law potential gives oldest universe, but only marginally.

Problem 12 The luminosity distance is given by

$$d_L = \frac{c(1+z)}{H_0} \int_0^z \frac{dz'}{H(z')/H_0}$$

We can change the integration variable from z to N again if we wish to.
See figure 6 for the plot.

Problem 13 Here it is a good idea to have stored the results from problem 12 so you just need to interpolate in them to find d_L at the required redshifts.
I found

$$\chi^2 = 34,5$$

for the power-law potential, and

$$\chi^2 = 98,4$$

for the exponential potential, so the power-law potential fits the data best (and the exponential potential is actually a horrible fit)

Problem 14

With just one parameter, Ω_{m0}, to optimize, a brute-force approach is feasible: Just calculate χ^2 for lots of values of Ω_{m0} between 0 and 1 and see which one gives the lowest χ^2.

I found

$$\chi^2_{min} = 29{,}7 \quad \text{for} \quad \Omega_{m0} = 0{,}30$$

This is a better fit than any of the two quintessence models. However, the comparison is not fair as I have found the value of Ω_{m0} which gives the best fit, whereas no optimization was done for the quintessence models. Maybe we could have found better values for the parameters in those models if we had tried.

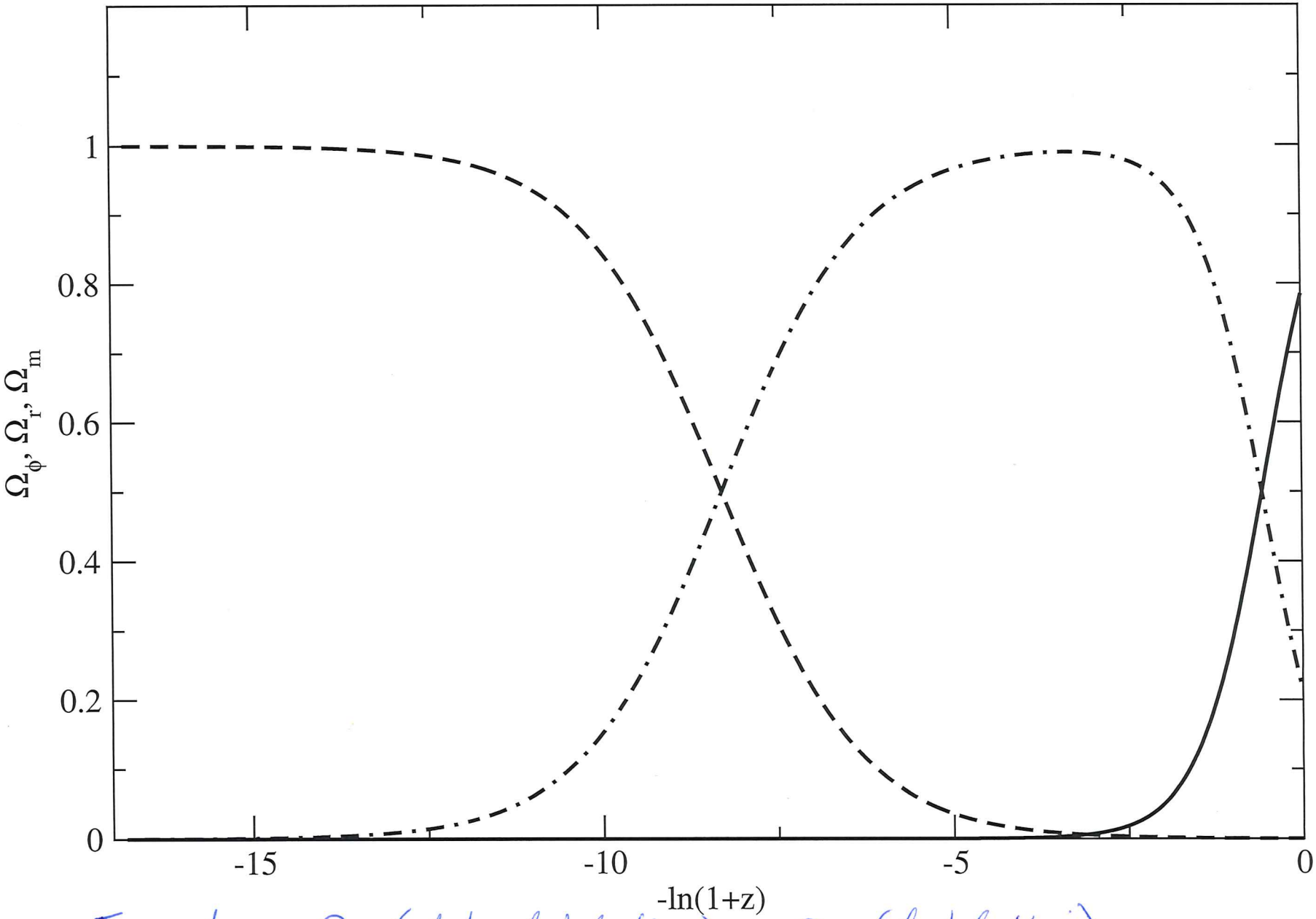

Figure 1: Ω_m (dot-dashed line), Ω_r (dashed line), and Ω_ϕ (full line) for the power-law potential

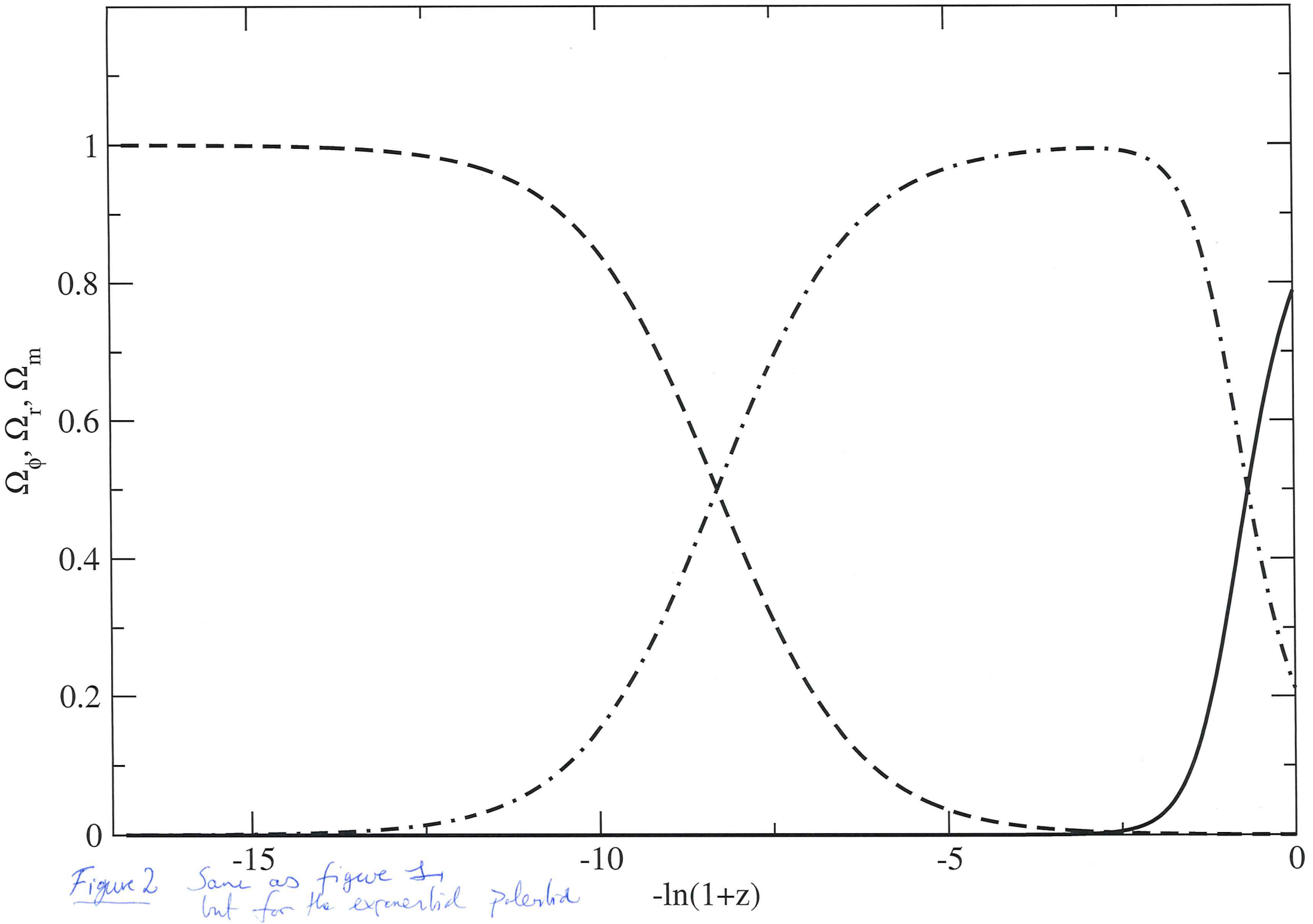

Figure 2 Same as figure 1 but for the exponential potential

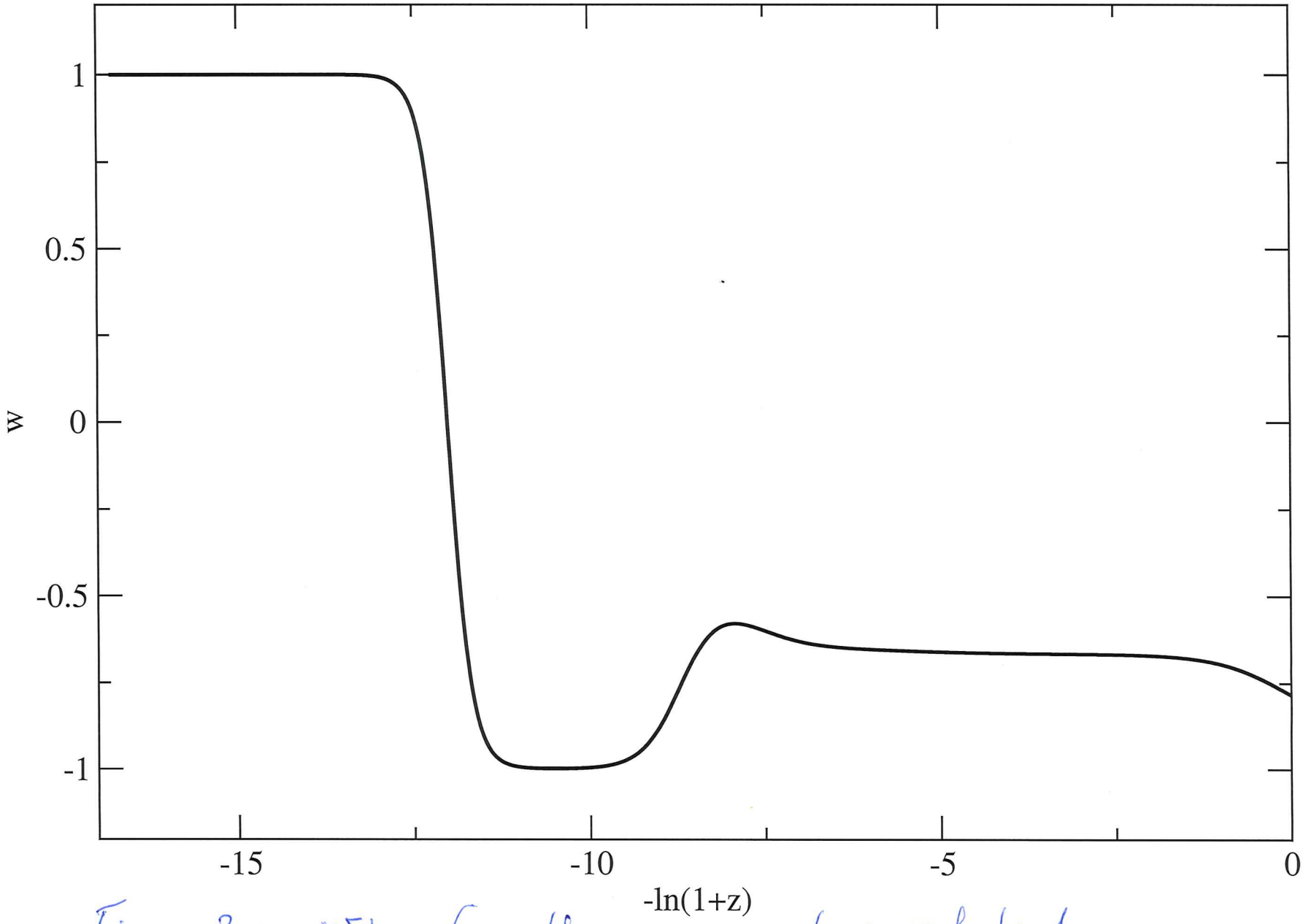

Figure 3: w_t for the power-law potential

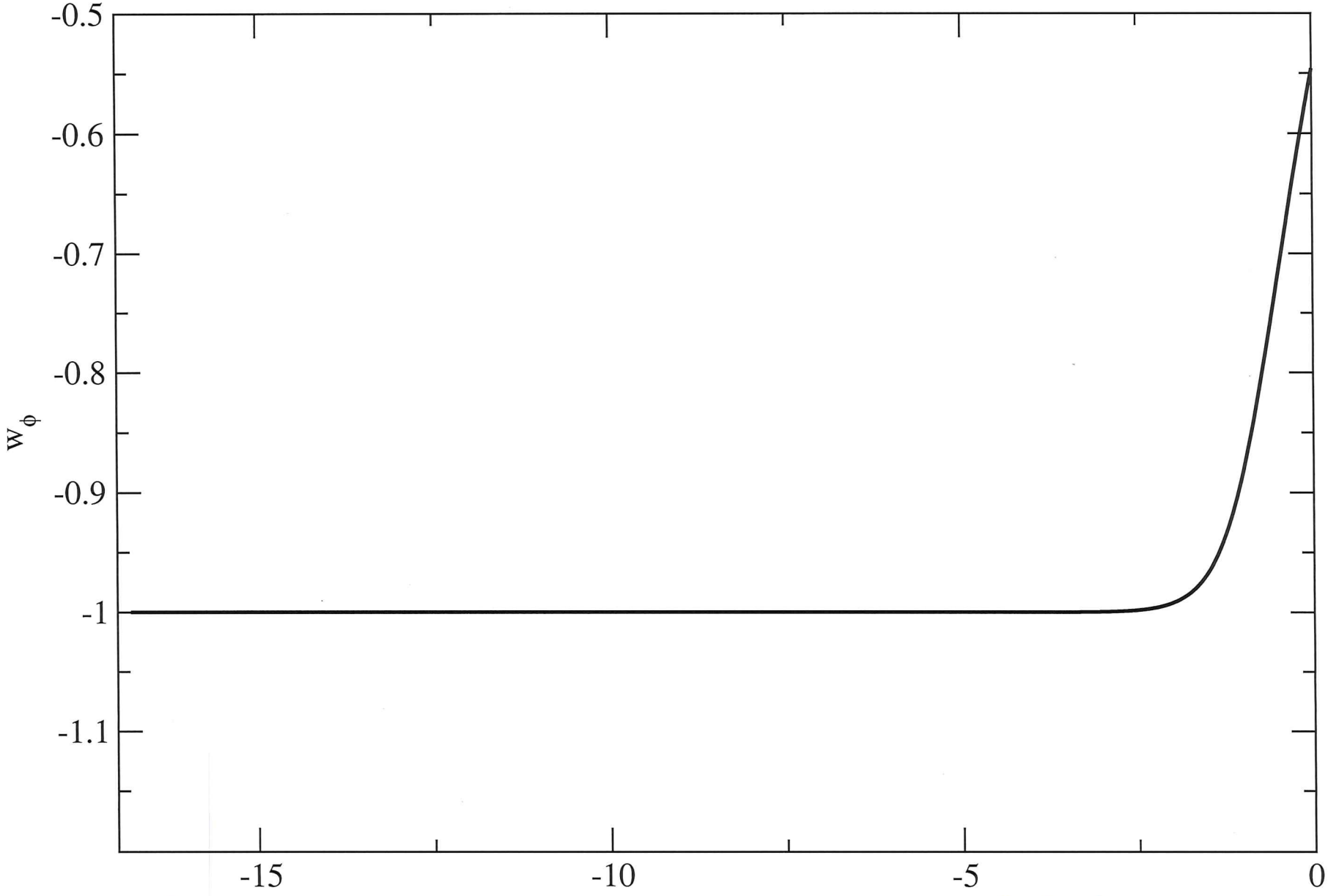

Figure 4 : w_ϕ for the exponential potential

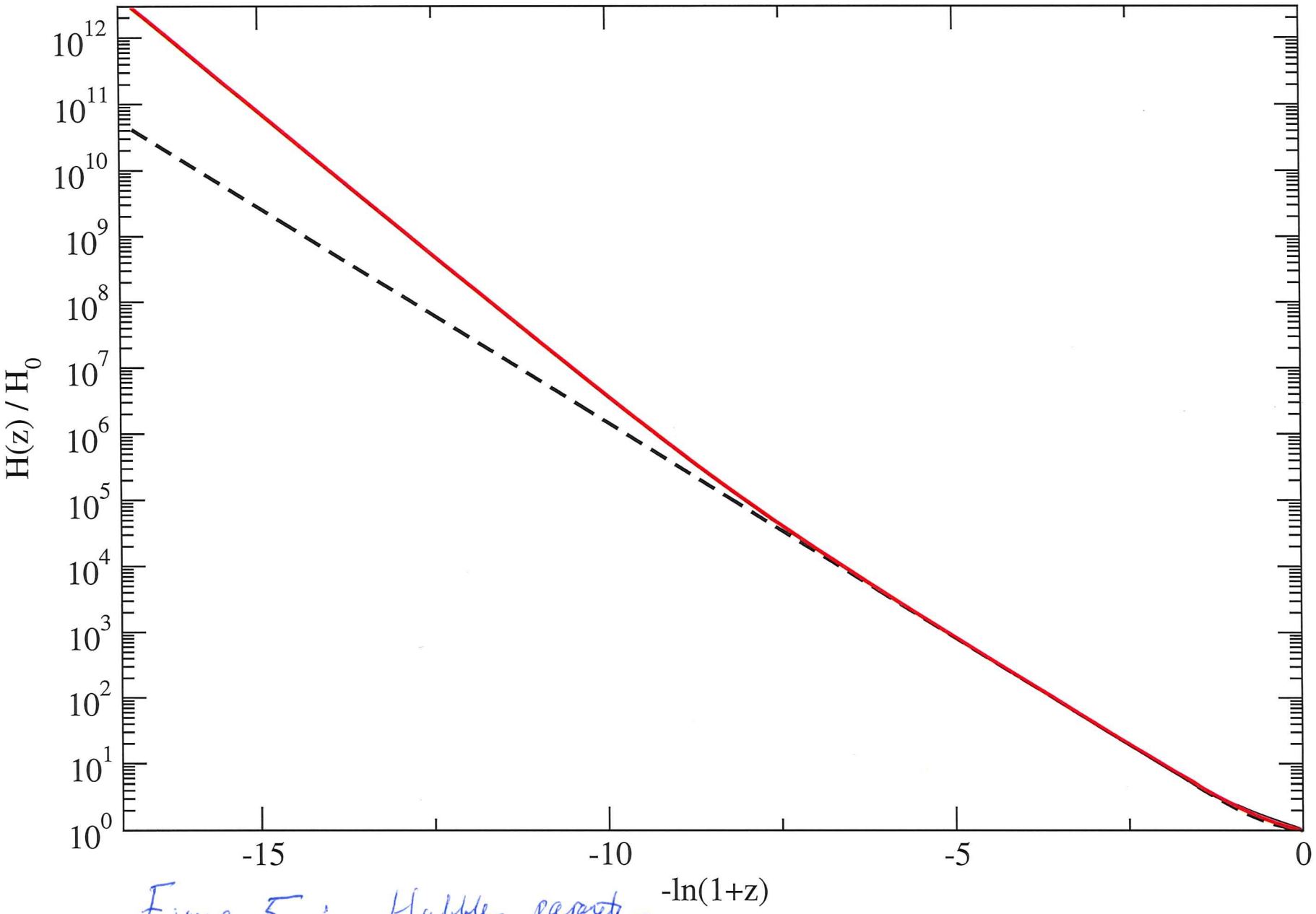

Figure 5 : Hubble parameter for power-law and exp. potentials (not visible separately) and the ΛCDM model (dashed line)

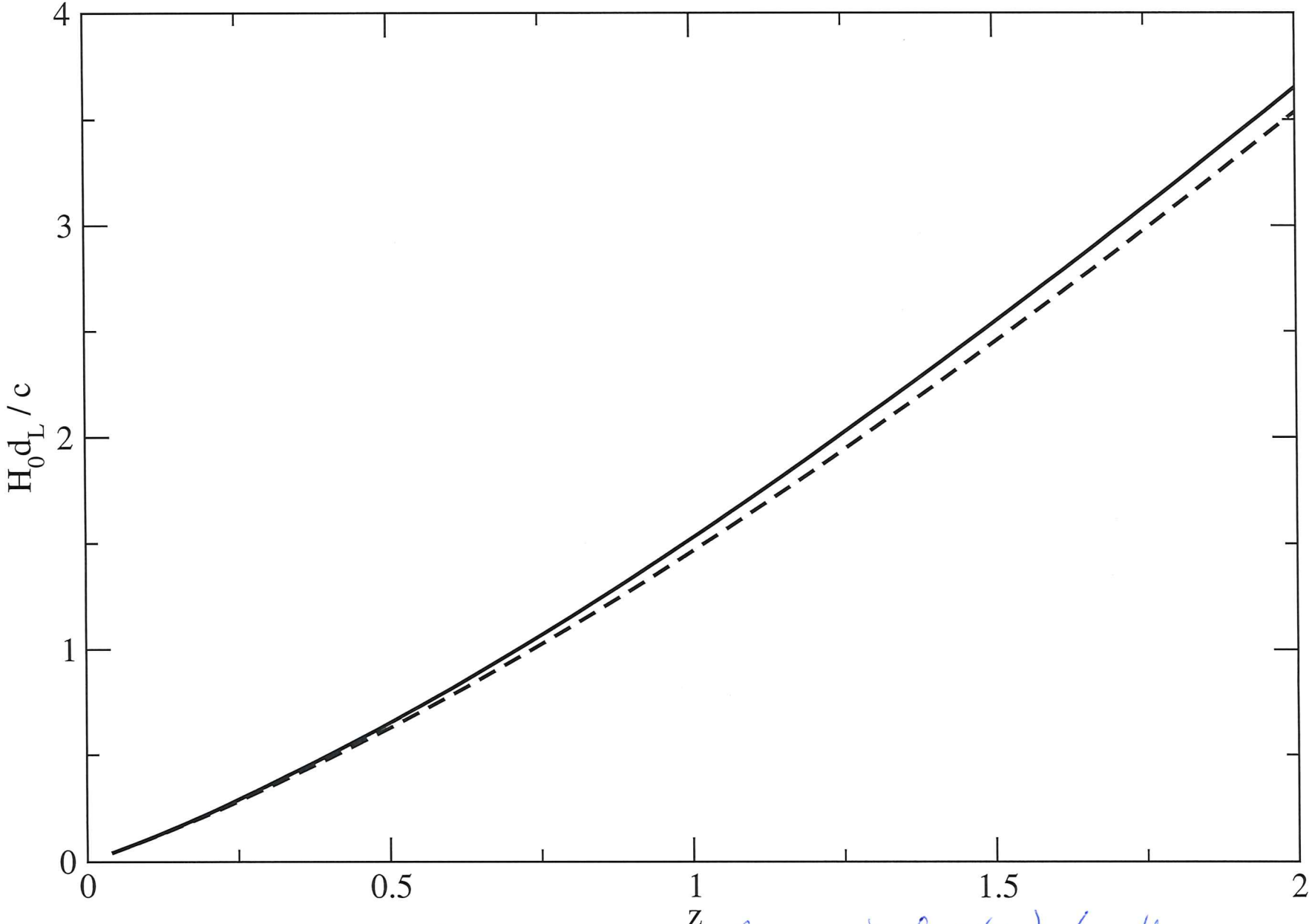

Figure 6: Luminosity distance (in units of c/H_0) for the power-law potential (full line) and the exponential potential (dashed line)

```
      program powerlaw_quintessence
      implicit none
      integer           np,nvar,N,M
      integer           nok,nbad,i,j,nint,nz,ndat
      parameter         (M=4,np=1000,nint=100,nz=50,ndat=31)
      double precision ystart(M),x1,x2,eps,h1,hmin
      double precision y(M,np+1),x(np+1)
      double precision ophi(np+1),or(np+1),om(np+1)
      double precision wphi(np+1)
      double precision xg(nint),dxg(nint)
      double precision ophi0,or0,om0
      double precision wint,Hubble(np+1),wp
      double precision HLCDM,H0t0,hub,cH0
      double precision term1,term2,term3
      double precision dL(nz),z(nz), Nlow
      double precision zdat(ndat),dLdat(ndat),err(ndat)
      double precision chi2,dLtheor
      external  derivs,rkqs

! Setting parameter values

      hub=0.7d0
      cH0=2.998d0/hub       ! in Gpc

!     Number of differential equations to solve

      nvar = M

!     Setting up grid where the solution is to be found

      do i = 1, np+1
         x(i)=-dlog(1.d0+2.d7)+dlog(1.d0+2.d7)*(i-1)/np
      end do

!     Initial conditons

      ystart(1) = 5.d-5
      ystart(2) = 1.d-8
      ystart(3) = 0.9999d0
      ystart(4) = 1.d9

      y(1,1) = ystart(1)
      y(2,1) = ystart(2)
      y(3,1) = ystart(3)
      y(4,1) = ystart(4)

!     Accuracy, step size and minimum step size, required by the ODE solver

      eps = 1.d-7
      h1 = 1.d-5
      hmin = 1.d-8

!     Loop over grid, find solutions on each grid point
```

```
      do i = 2, np+1
         x1 = x(i-1)
         x2 = x(i)
         call odeint(ystart,nvar,x1,x2,eps,h1,hmin,nok,
     *        nbad,derivs,rkqs)

         do j = 1, nvar
            y(j,i) = ystart(j)
         end do
      end do

!     Generate data for plots

      do i = 1, np+1
         ophi(i)=y(1,i)**2+y(2,i)**2
         or(i) = y(3,i)**2
         om(i) = 1.d0-ophi(i)-or(i)
         wphi(i)= (y(1,i)**2-y(2,i)**2)/ophi(i)
      end do

      open(8,file='densities_power_law.dat')
      open(9,file='EoS_power_law.dat')
      open(10,file='weff_power_law.dat')
      do i = 1, np+1
         write(8,*)x(i),ophi(i),or(i),om(i)
         write(9,*)x(i),wphi(i)
         write(10,*)x(i),y(1,i)**2-y(2,i)**2+y(3,i)**2/3.d0
      end do
      close(8)
      close(9)
      close(10)


!     Calculating the Hubble parameter

      ophi0=ophi(np+1)
      or0=or(np+1)
      om0=om(np+1)

      do i = 1, np+1
         wint=0.d0
         call gauleg(x(i),0.d0,xg,dxg,nint)
         do j = 1,nint
            call interp1d(x,wphi,np+1,3,xg(j),wp)
            wint=wint+(1.d0+wp)*dxg(j)
         end do
         term1=or0*dexp(-4.d0*x(i))
         term2=om0*dexp(-3.d0*x(i))
         term3=ophi0*dexp(3.d0*wint)
         hubble(i)=dsqrt(term1+term2+term3)
      end do

!     Output results for plotting

      open(8,file='hubble_power_law.dat')
```

```
      do i = 1, np+1
         write(8,*)x(i),hubble(i)
      end do
      close(8)

!     Calculating the Hubble parameter in LCDM for comparison

      open(8,file='hubble_LCDM.dat')
      do i = 1,np+1
         term1=om0*dexp(-3.d0*x(i))
         term2=1.d0-om0
         HLCDM=dsqrt(term1+term2)
         write(8,*)x(i),HLCDM
      end do
      close(8)

!     Calculating the age

      H0t0=0.d0
      call gauleg(x(1),x(np+1),xg,dxg,nint)
      do i = 1, nint
         call interp1d(x,hubble,np+1,3,xg(i),term1)
         H0t0=H0t0+dxg(i)/term1
      end do
      write(*,*)'H0*t0 = ', H0t0

!     Calculating luminosity distance

      do i = 1, nz
         z(i) = 2.d0*i/nz
         Nlow=-dlog(1.d0+z(i))
         dL(i) = 0.d0
         call gauleg(Nlow,0.d0,xg,dxg,nint)
         do j = 1, nint
            call interp1d(x,hubble,np+1,3,xg(j),term1)
            dL(i)=dL(i)+dexp(-xg(j))*dxg(j)/term1
         end do
         dL(i) = dL(i)*dexp(-Nlow)
      end do

!     Output of results for plotting

      open(8,file='dL_power_law.dat')
      do i = 1, nz
         write(8,*)z(i),dL(i)
      end do
      close(8)

!     Loading SN1a data

      open(8,file='sndata.txt')
      do i = 1, ndat
         read(8,*)zdat(i),dLdat(i),err(i)
      end do
      close(8)
```

```
!     Calculating chi^2

      chi2 = 0.d0
      open(8,file='expcompare_power_law.dat')
      do  i = 1, ndat
         call interp1d(z,dL,nz,3,zdat(i),dLtheor)
         dLtheor=dLtheor*cH0
         write(8,*)zdat(i),dLtheor
         chi2=chi2+(dLtheor-dLdat(i))**2/err(i)**2
      end do
      close(8)
      write(*,*)chi2

      end



!     The following routines make up the ODE solver, and they are taken from
!      "Numerical Recipes in FORTRAN" by Press, Teukolsky, Vetterling and
!      Flannery

      subroutine derivs(x,y,dydx)
      implicit none
      double precision  x,y(4),dydx(4)
      double precision  s62,term

      s62=dsqrt(6.d0)/2.d0
      term = 3.d0+3.d0*y(1)*y(1)-3.d0*y(2)*y(2)+y(3)*y(3)

      dydx(1)=-3.d0*y(1)+s62*y(4)*y(2)*y(2)+.5d0*y(1)*term
      dydx(2)=-s62*y(4)*y(1)*y(2)+.5d0*y(2)*term
      dydx(3)=-2.d0*y(3)+.5d0*y(3)*term
      dydx(4)=-2.d0*s62*y(4)*y(4)*y(1)

      return
      end




      SUBROUTINE odeint(ystart,nvar,x1,x2,eps,h1,hmin,nok,nbad,derivs,
     *rkqs)
      INTEGER nbad,nok,nvar,KMAXX,MAXSTP,NMAX
      double precision eps,h1,hmin,x1,x2,ystart(nvar),TINY
      EXTERNAL derivs,rkqs
      PARAMETER (MAXSTP=10000,NMAX=50,KMAXX=200,TINY=1.d-30)
      INTEGER i,kmax,kount,nstp
      double precision dxsav,h,hdid,hnext,x,xsav,dydx(NMAX),xp(KMAXX),
     *y(NMAX),
     *yp(NMAX,KMAXX),yscal(NMAX)
      COMMON /path/ kmax,kount,dxsav,xp,yp
      x=x1
      h=sign(h1,x2-x1)
```

```
      nok=0
      nbad=0
      kount=0
      do 11 i=1,nvar
        y(i)=ystart(i)
11    continue
      if (kmax.gt.0) xsav=x-2.d0*dxsav
      do 16 nstp=1,MAXSTP
        call derivs(x,y,dydx)
        do 12 i=1,nvar
          yscal(i)=dabs(y(i))+dabs(h*dydx(i))+TINY
12      continue
        if(kmax.gt.0)then
          if(dabs(x-xsav).gt.dabs(dxsav)) then
            if(kount.lt.kmax-1)then
              kount=kount+1
              xp(kount)=x
              do 13 i=1,nvar
                yp(i,kount)=y(i)
13            continue
              xsav=x
            endif
          endif
        endif
        if((x+h-x2)*(x+h-x1).gt.0.d0) h=x2-x
        call rkqs(y,dydx,nvar,x,h,eps,yscal,hdid,hnext,derivs)
        if(hdid.eq.h)then
          nok=nok+1
        else
          nbad=nbad+1
        endif
        if((x-x2)*(x2-x1).ge.0.d0)then
          do 14 i=1,nvar
            ystart(i)=y(i)
14        continue
          if(kmax.ne.0)then
            kount=kount+1
            xp(kount)=x
            do 15 i=1,nvar
              yp(i,kount)=y(i)
15          continue
          endif
          return
        endif
        if(dabs(hnext).lt.hmin) pause
     *'stepsize smaller than minimum in odeint'
        h=hnext
16    continue
      pause 'too many steps in odeint'
      return
      END
C  (C) Copr. 1986-92 Numerical Recipes Software Vs1&v%1jw#<?4210(9p#.


      SUBROUTINE rkck(y,dydx,n,x,h,yout,yerr,derivs)
```

```
      INTEGER n,NMAX
      double precision h,x,dydx(n),y(n),yerr(n),yout(n)
      EXTERNAL derivs
      PARAMETER (NMAX=50)
CU    USES derivs
      INTEGER i
      double precision ak2(NMAX),ak3(NMAX),ak4(NMAX)
      double precision ak5(NMAX),ak6(NMAX)
      double precision ytemp(NMAX),A2,A3,A4,A5,A6,B21,B31,B32,B41
      double precision B42,B43,B51,B52,B53
      double precision B54,B61,B62,B63,B64,B65,C1,C3
      double precision  C4,C6,DC1,DC3,DC4,DC5,DC6
      PARAMETER (A2=.2d0,A3=.3d0,A4=.6d0,
     *A5=1.d0,A6=.875d0,B21=.2d0,B31=3.d0/40.d0,
     *B32=9.d0/40.d0,B41=.3d0,B42=-.9d0,
     *B43=1.2d0,B51=-11.d0/54.d0,B52=2.5d0,
     *B53=-70.d0/27.d0,B54=35.d0/27.d0,
     *B61=1631.d0/55296.d0,B62=175.d0/512.d0,
     *B63=575.d0/13824.d0,B64=44275.d0/110592.d0,
     *B65=253.d0/4096.d0,C1=37.d0/378.d0,
     *C3=250.d0/621.d0,C4=125.d0/594.d0,
     *C6=512.d0/1771.d0,DC1=C1-2825.d0/27648.d0,
     *DC3=C3-18575.d0/48384.d0,
     *DC4=C4-13525.d0/55296.d0,DC5=-277.d0/14336.d0,
     *DC6=C6-.25d0)
      do 11 i=1,n
        ytemp(i)=y(i)+B21*h*dydx(i)
11    continue
      call derivs(x+A2*h,ytemp,ak2)
      do 12 i=1,n
        ytemp(i)=y(i)+h*(B31*dydx(i)+B32*ak2(i))
12    continue
      call derivs(x+A3*h,ytemp,ak3)
      do 13 i=1,n
        ytemp(i)=y(i)+h*(B41*dydx(i)+B42*ak2(i)+B43*ak3(i))
13    continue
      call derivs(x+A4*h,ytemp,ak4)
      do 14 i=1,n
        ytemp(i)=y(i)+h*(B51*dydx(i)+B52*ak2(i)+B53*ak3(i)+B54*ak4(i))
14    continue
      call derivs(x+A5*h,ytemp,ak5)
      do 15 i=1,n
        ytemp(i)=y(i)+h*(B61*dydx(i)+B62*ak2(i)+B63*ak3(i)+B64*ak4(i)+
     *B65*ak5(i))
15    continue
      call derivs(x+A6*h,ytemp,ak6)
      do 16 i=1,n
        yout(i)=y(i)+h*(C1*dydx(i)+C3*ak3(i)+C4*ak4(i)+C6*ak6(i))
16    continue
      do 17 i=1,n
        yerr(i)=h*(DC1*dydx(i)+DC3*ak3(i)+DC4*ak4(i)+DC5*ak5(i)+DC6*
     *ak6(i))
17    continue
      return
      END
```

```
C  (C) Copr. 1986-92 Numerical Recipes Software Vs1&v%1jw#<?4210(9p#.


      SUBROUTINE rkqs(y,dydx,n,x,htry,eps,yscal,hdid,hnext,derivs)
      INTEGER n,NMAX
      double precision eps,hdid,hnext,htry,x,dydx(n),y(n),yscal(n)
      EXTERNAL derivs
      PARAMETER (NMAX=50)
CU    USES derivs,rkck
      INTEGER i
      double precision errmax,h,xnew,yerr(NMAX),ytemp(NMAX),SAFETY,PGROW,PSHRNK,
     *ERRCON
      PARAMETER (SAFETY=0.9d0,PGROW=-.2d0,PSHRNK=-.25d0,
     *ERRCON=1.89d-4)
      h=htry
1     call rkck(y,dydx,n,x,h,ytemp,yerr,derivs)
      errmax=0.d0
      do 11 i=1,n
        errmax=max(errmax,dabs(yerr(i)/yscal(i)))
11    continue
      errmax=errmax/eps
      if(errmax.gt.1.d0)then
        h=SAFETY*h*(errmax**PSHRNK)
        if(h.lt.0.1d0*h)then
          h=.1d0*h
        endif
        xnew=x+h
        if(xnew.eq.x)pause 'stepsize underflow in rkqs'
        goto 1
      else
        if(errmax.gt.ERRCON)then
          hnext=SAFETY*h*(errmax**PGROW)
        else
          hnext=5.d0*h
        endif
        hdid=h
        x=x+h
        do 12 i=1,n
          y(i)=ytemp(i)
12      continue
        return
      endif
      END
C  (C) Copr. 1986-92 Numerical Recipes Software Vs1&v%1jw#<?4210(9p#.


! Routines for interpolation, from the same source


      SUBROUTINE interp1d(x,fx,np,npol,x1,fx1)
!-------------------------------------------------------------------------
!     input arrays: x(np),fx(np)   ->FUNCTION f(x)
!           value  x1 the point
```

```
!     output value   fx1           -> f(x1)
!     mp: x(mp)< x1 <x(mp+1)
!     mp=1 IF x1 < x(1)
!     mp=np IF x1 > x(np)
!     npol= # of interpolation points
!-----------------------------------------------------------------------
      IMPLICIT REAL*8(a-h,o-z)
      REAL*8 x(np),fx(np),x1,fx1,df
      INTEGER mp,np,l,npol
      IF(x1.EQ.x(1))THEN
        fx1=fx(1)
      ELSEIF(x1.EQ.x(np))THEN
        fx1=fx(np)
      ELSE
       CALL LOCATE(x,NP,x1,MP)
       L=MIN(MAX(MP-(npol-1)/2,1),NP+1-npol)
!       IF(mp.EQ.0.OR.mp.EQ.np)THEN
!         IF(mp.EQ.0)THEN
!            WRITE(6,*)'warning x1= ',x1,' and  x(',1,')= ',x(1)
!         ELSE
!            WRITE(6,*)'warning x1= ',x1,' and  x(',np,')= ',x(np)
!         ENDIF
!         WRITE(6,*)np,npol
!         DO i=1,np
!            WRITE(6,*)i,x(i),fx(i)
!         ENDDO
!       ENDIF
       CALL POLINT(x(L),fx(L),3,x1,fx1,df)
      ENDIF
      RETURN
      END



      SUBROUTINE locate(xx,n,x,j)
      IMPLICIT REAL*8(A-H,O-Z)
      INTEGER n,j,jl,ju,jm
      REAL*8 x,xx(n)
      jl=0
      ju=n+1
10    IF(ju-jl.GT.1) THEN
       jm=(ju+jl)/2
       IF((xx(n).GT.xx(1)).EQV.(x.GT.xx(jm))) THEN
          jl=jm
       ELSE
          ju=jm
       ENDIF
       GOTO 10
      ENDIF
      j=jl
      RETURN
      END
```

```
      SUBROUTINE polint(xa,ya,n,x,y,dy)
      INTEGER n,NMAX
      double precision dy,x,y,xa(n),ya(n)
      PARAMETER (NMAX=10)
      INTEGER i,m,ns
      double precision den,dif,dift,ho,hp,w,c(NMAX),d(NMAX)
      ns=1
      dif=dabs(x-xa(1))
      do 11 i=1,n
        dift=dabs(x-xa(i))
        if (dift.lt.dif) then
          ns=i
          dif=dift
        endif
        c(i)=ya(i)
        d(i)=ya(i)
11    continue
      y=ya(ns)
      ns=ns-1
      do 13 m=1,n-1
        do 12 i=1,n-m
          ho=xa(i)-x
          hp=xa(i+m)-x
          w=c(i+1)-d(i)
          den=ho-hp
          if(den.eq.0.d0)pause 'failure in polint'
          den=w/den
          d(i)=hp*den
          c(i)=ho*den
12      continue
        if (2*ns.lt.n-m)then
          dy=c(ns+1)
        else
          dy=d(ns)
          ns=ns-1
        endif
        y=y+dy
13    continue
      return
      END

! Routine for Gauss-Legendre integration, also from the same source

      SUBROUTINE gauleg(x1,x2,x,w,n)
      INTEGER n
      double precision x1,x2,x(n),w(n)
      DOUBLE PRECISION EPS
      PARAMETER (EPS=3.d-14)
      INTEGER i,j,m
      DOUBLE PRECISION p1,p2,p3,pp,xl,xm,z,z1

      m=(n+1)/2
      xm=0.5d0*(x2+x1)
      xl=0.5d0*(x2-x1)
      do  i=1,m
```

```
      z=dcos(3.141592654d0*(i-.25d0)/(n+.5d0))
1     continue
        p1=1.d0
        p2=0.d0
        do  j=1,n
          p3=p2
          p2=p1
          p1=((2.d0*j-1.d0)*z*p2-(j-1.d0)*p3)/j
       end do
        pp=n*(z*p1-p2)/(z*z-1.d0)
        z1=z
        z=z1-p1/pp
      if(dabs(z-z1).gt.EPS)goto 1
      x(i)=xm-xl*z
      x(n+1-i)=xm+xl*z
      w(i)=2.d0*xl/((1.d0-z*z)*pp*pp)
      w(n+1-i)=w(i)
    end do
    return
    END
```