Astrophysics V Observational Cosmology

Sheet 7: Solutions

Prof. Jean-Paul Kneib

Teaching assistants: Dr. Rafaela Gsponer, Dr. Antoine Rocher, Mathilde Guitton, Shengyu He, Ashutosh Mishra & Aurélien Verdier

Laboratoire d'astrophysique http://lastro.epfl.ch
Ecole Polytechnique Fédérale de Lausanne, Spring Semester 2025

Exercise 1 : Large scale galaxy spectroscopic surveys

To calculate the volume of the survey in the comoving space, one can imagine a spherical shell around the Earth, with the inner radius $d_{\rm com}(z_{\rm min})$ and the outer radius $d_{\rm com}(z_{\rm max})$, where $z_{\rm min} = 0.2$ and $z_{\rm max} = 0.75$. The volume of such shell is :

$$\frac{4\pi}{3} \times [d_{\rm com}(z_{\rm max})^3 - d_{\rm com}(z_{\rm min})^3].$$

This is not the volume covered by the survey, because a spherical shell implies the coverage of the whole sky. However, BOSS has covered only 9329 deg^2. Thus, one should compute the area of the entire sky which is 4π steradians or

$$4\pi \left(\frac{180}{\pi}\right)^2 \text{deg}^2.$$

Now, one can compute the fraction of the sky covered by BOSS and multiply it with the volume of the spherical shell, leading to comoving volume for LRGs of $5.78383 \times 10^9\,\text{Mpc}^3$. Finally, the number density of BOSS LRGs is $0.207475 \times 10^{-3}\,h^3\text{Mpc}^{-3}$.

The largest separation between two galaxies is given by transverse comoving distance at $z_{\rm max}$ times the maximum angle separation between galaxies which is 150 deg. Numerically, this distance is $\approx 4833\,\text{Mpc}$.

One can imagine that this separation occurs along the diagonal of the simulation box, thus the side length of the simulation box should be the length of the diagonal divided by the square root of 3. Numerically, this distance is $\approx 2790\,\text{Mpc}$

[breakable, size=fbox, boxrule=1pt, pad at break*=1mm,colback=cellbackground, colframe=cellborder] Inincolor1

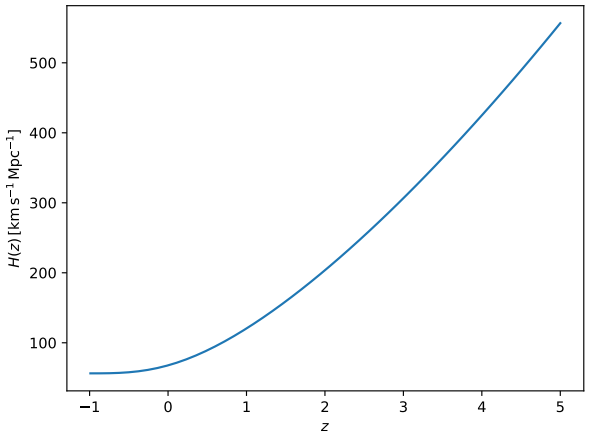
```
# Mean number density of BOSS LRGs
from nbodykit.lab import cosmology
import numpy as np

num = 1.2e6
area = 9329
zmin = 0.2
zmax = 0.75
angle_max = 150

# The ratio of the BOSS survey area to the full sky area
sky_area = 4 * np.pi * (180.0 / np.pi)**2
area_frac = area / sky_area

# The range of the radial comoving distance for BOSS LRGs
cosmo = cosmology.Planck15
dC_min = cosmo.comoving_distance(zmin)
dC_max = cosmo.comoving_distance(zmax)

# The comoving volume of BOSS LRGs
sky_vol = 4.0 / 3.0 * np.pi * (dC_max**3 - dC_min**3)
volume = sky_vol * area_frac
print('Comoving volume of BOSS LRGs: ' \
      '{0:g} Mpc^3 h^{{-3}}'.format(volume))

# The mean number density of BOSS LRGs
dens = num / volume
print('Mean number density of BOSS LRGs: ' \
      '{0:g} h^3 Mpc^{{-3}}'.format(dens))

# The maximum transverse comoving distance
# between two galaxies
angle = angle_max * np.pi / 180.0
d_CT = angle * cosmo.comoving_transverse_distance(zmax)

# The minimum side length of the box covering all BOSS
# data, if aligning galaxies with the largest separation
# along the diagonal direction.
L_sim = d_CT / np.sqrt(3)
print('Minimum side length of the simulation ' \
```

```
      'box for BOSS LRGs:\n  {0:g} Mpc/h'.format(L_sim))
```

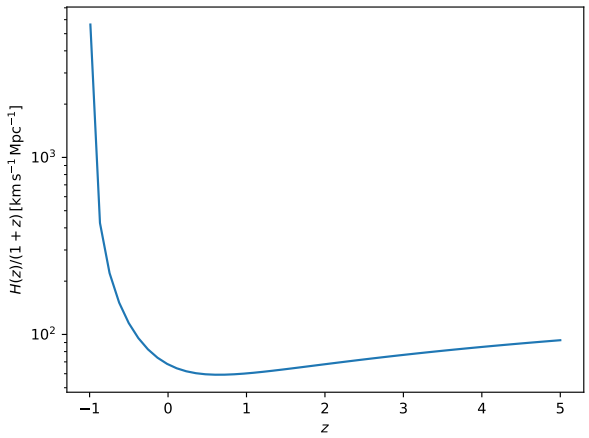
```
Comoving volume of BOSS LRGs: 5.78383e+09 Mpc^3 h^{-3}
Mean number density of BOSS LRGs: 0.000207475 h^3 Mpc^{-3}
Minimum side length of the simulation box for BOSS LRGs:
  2790.43 Mpc/h
1275.1839653815382
```

Exercise 2 : Hubble parameter and expansion rate

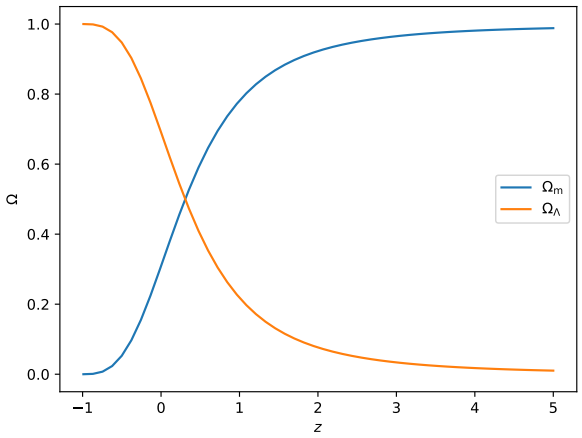
```
In [1]: # Evolution of Hubble parameter
        from nbodykit.lab import cosmology
        import numpy as np
        import matplotlib.pyplot as plt
        %matplotlib inline

        cosmo = cosmology.Planck15

        amax = 100.0
        cosmo = cosmo.clone(a_max=amax)
        z = np.linspace(1.0/amax-1,5,50)

        H = cosmo.efunc(z) * cosmo.h * 100
        plt.plot(z, H)
        plt.xlabel(r'$z$')
        plt.ylabel(r'$H(z)\,[{\rm km}\,{\rm s}^{-1}\,' \
                r'{\rm Mpc}^{-1}]$')
        plt.show()
```




```
In [2]: # The expansion rate of the Universe
        plt.plot(z, H/(1+z))
        plt.yscale('log')
        plt.xlabel(r'$z$')
        plt.ylabel(r'$H(z) / (1+z) ' \
                r'\,[{\rm km}\,{\rm s}^{-1}\,{\rm Mpc}^{-1}]$')
        plt.show()
```



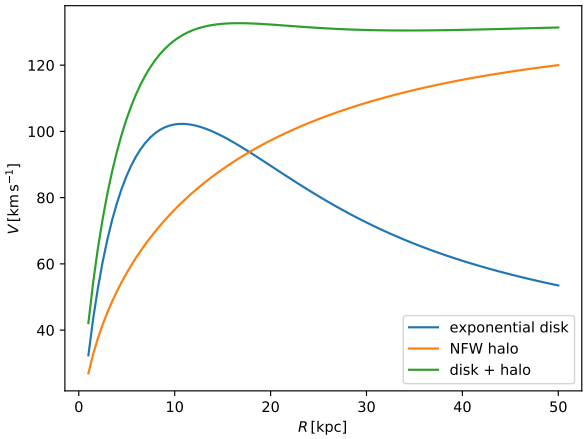
In [3]:
```
# Evolution of components of the Universe
rho_m = cosmo.rho_m(z)
rho_l = cosmo.rho_lambda(z)
rho_c = cosmo.rho_crit(z)

Om = rho_m / rho_c
OL = rho_l / rho_c

plt.plot(z, Om, label=r'$\Omega_{\rm m}$')
plt.plot(z, OL, label=r'$\Omega_\Lambda$')

plt.xlabel(r'$z$')
plt.ylabel(r'$\Omega$')
plt.legend()
plt.show()
```



Exercise 3 : Rotation curve of spiral galaxies

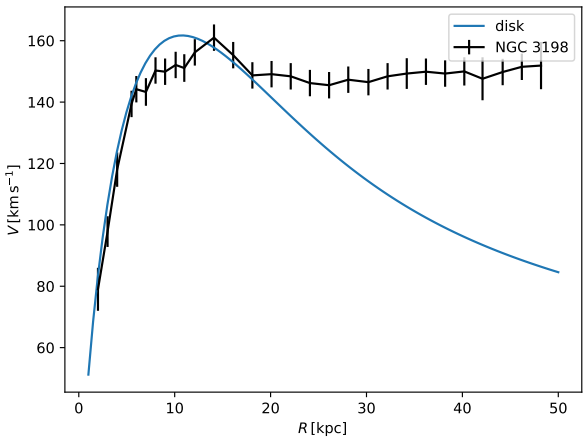
In [1]:
```
# Rotation curve of exponential disk and NFW halo
import matplotlib.pyplot as plt
%matplotlib inline
import numpy as np
import scipy.special as sp
```

```
def v2disk(Sigma0, Rd, R):
    y = R * 0.5 / Rd
    G = 4.302e-6    # in kpc (km/s)^2 / M_solar
    fac = sp.iv(0,y) * sp.kn(0,y) - sp.iv(1,y) * sp.kn(1,y)
    return 4 * np.pi * G * Sigma0 * Rd * y**2 * fac

def v2halo(V200, R200, Rs, R):
    c = R200 / Rs
    x = R / R200
    cx = c * x
    fac = np.log(1 + cx) - cx / (1 + cx)
    fac /= (np.log(1 + c) - c / (1 + c))
    return V200**2 / x * fac

R0 = np.linspace(1, 50, 100)
Vd2 = v2disk(2e8, 5, R0)
Vh2 = v2halo(120, 50, 50, R0)

Vd = np.sqrt(Vd2)
Vh = np.sqrt(Vh2)
Vtot = np.sqrt(Vd2 + Vh2)

plt.plot(R0, Vd, label='exponential disk')
plt.plot(R0, Vh, label='NFW halo')
plt.plot(R0, Vtot, label='disk + halo')
plt.xlabel(r'$R\,[{\rm kpc}]$')
plt.ylabel(r'$V\,[{\rm km}\,{\rm s}^{-1}]$')
plt.legend()
plt.show()
```



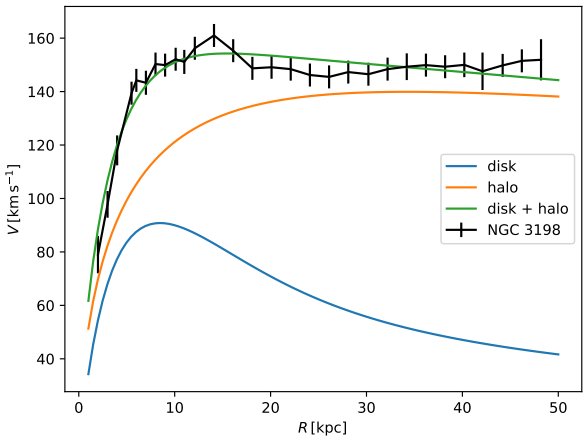
In [2]:
```
# Rotation curve of NGC 3198 (arXiv:1503.04049)
R = [ 2. ,  3. ,  4. ,  5.5,  6. ,  7. ,  8. , \
      9. , 10.1, 11. , 12.1, 14.1, 16.1, 18.1, \
     20.1, 22.1, 24.1, 26.1, 28.1, 30.2, 32.2, \
     34.2, 36.2, 38.2, 40.2, 42.1, 44.2, 46.2, 48.2]
V = [ 79. ,  97.8, 118. , 139.4, 144.2, 143.3, \
     150.3, 149.9, 152.1, 151.1, 156.2, 161. , \
     155.3, 148.7, 149.1, 148.4, 146.2, 145.5, \
     147.3, 146.5, 148.4, 149.3, 149.9, 149.3, \
     150. , 147.6, 149.8, 151.5, 151.9]
dV = [7. , 5. , 5.6, 4.3, 4.3, 4.5, 4.3, 4.3, \
      4.3, 4.5, 4.3, 4.3, 4.3, 4.3, 4.3, 4.3, \
      4.3, 4.3, 4.3, 4.3, 4.3, 5. , 4.3, 4.3, \
      4.6, 7. , 4.3, 4.3, 7.7]

R = np.array(R)
V = np.array(V)
dV = np.array(dV)
Vd = np.sqrt(v2disk(5e8, 5, R0))

plt.errorbar(R, V, yerr=dV, color='k', label='NGC 3198')
plt.plot(R0, Vd, label='disk')

plt.xlabel(r'$R\,[{\rm kpc}]$')
```

```
plt.ylabel(r'$V\,[{\rm km}\,{\rm s}^{-1}]$')
plt.legend()
plt.show()
```



In [3]:
```
# Fitting of rotation curve
from scipy.optimize import minimize

def chi2(par, R, V, dV):
    Sigma0, Rd = par[0], par[1]
    V200, R200, Rs = par[2], par[3], par[4]
    Vd2 = v2disk(Sigma0, Rd, R)
    Vh2 = v2halo(V200, R200, Rs, R)
    Vtot = np.sqrt(Vd2 + Vh2)
    diff = ((V - Vtot) / dV)**2
    return diff.sum()

ini = [2e8, 5., 120., 50., 50.]
res = minimize(chi2, ini, args=(R,V,dV))

par = res.x
Sigma0, Rd = par[0], par[1]
V200, R200, Rs = par[2], par[3], par[4]
Vd2 = v2disk(Sigma0, Rd, R0)
Vh2 = v2halo(V200, R200, Rs, R0)
```

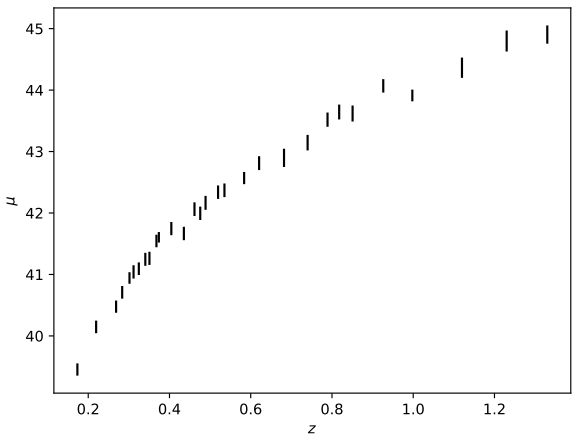
```
Vtot = np.sqrt(Vd2 + Vh2)
Vd = np.sqrt(Vd2)
Vh = np.sqrt(Vh2)

plt.errorbar(R, V, yerr=dV, color='k', label='NGC 3198')
plt.plot(R0, Vd, label='disk')
plt.plot(R0, Vh, label='halo')
plt.plot(R0, Vtot, label='disk + halo')

plt.xlabel(r'$R\,[{\rm kpc}]$')
plt.ylabel(r'$V\,[{\rm km}\,{\rm s}^{-1}]$')
plt.legend()
plt.show()
```



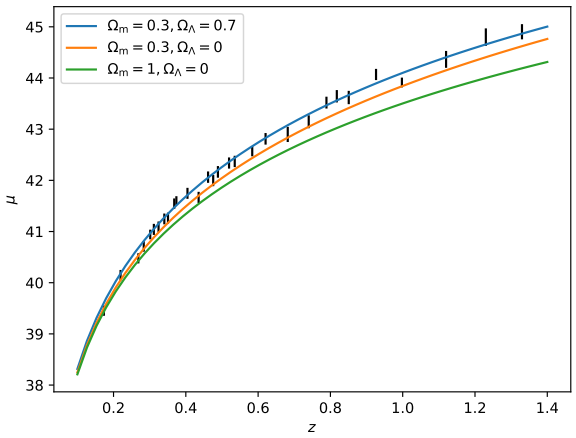
Exercise 4 : Cosmology with Type Ia supernovae

```
In [1]: # Distance modulus of Type Ia supernovae (arXiv:1710.00845)
        import matplotlib.pyplot as plt
        %matplotlib inline
        import numpy as np

        z = [0.17331, 0.21946, 0.26862, 0.2836 , 0.30158, \
             0.31156, 0.32454, 0.34052, 0.3507 , 0.36791, \
```

```
      0.37386, 0.40455, 0.43544, 0.46156, 0.47572, \
      0.48881, 0.51961, 0.53532, 0.58375, 0.62063, \
      0.68196, 0.74   , 0.78904, 0.81773, 0.85073, \
      0.92624, 0.99781, 1.12   , 1.23   , 1.33   ]
mB = [20.15345, 20.8469 , 21.1762 , 21.40915, 21.64245, \
       21.7417 , 21.79365, 21.9463 , 21.9623 , 22.24455, \
       22.3047 , 22.44625, 22.36675, 22.76265, 22.6952 , \
       22.8653 , 23.03865, 23.06945, 23.2686 , 23.5114 , \
       23.59755, 23.8446 , 24.2188 , 24.3431 , 24.31865, \
       24.7687 , 24.6122 , 25.0639 , 25.4985 , 25.6034 ]
dmB = [0.09925, 0.1023 , 0.0993 , 0.1028 , 0.0939 , \
        0.10825, 0.10225, 0.10555, 0.10725, 0.1019 , \
        0.0854 , 0.1075 , 0.109  , 0.11115, 0.109  , \
        0.1136 , 0.109  , 0.11085, 0.09915, 0.1131 , \
        0.1484 , 0.12605, 0.11585, 0.12085, 0.1304 , \
        0.10935, 0.0953 , 0.1649 , 0.17085, 0.1484 ]

MB = -19.3
z = np.array(z)
muB = np.array(mB) - MB
dmB = np.array(dmB)

plt.errorbar(z, muB, yerr=dmB, color='k', ls='None')
plt.xlabel(r'$z$')
plt.ylabel(r'$\mu$')
plt.show()
```



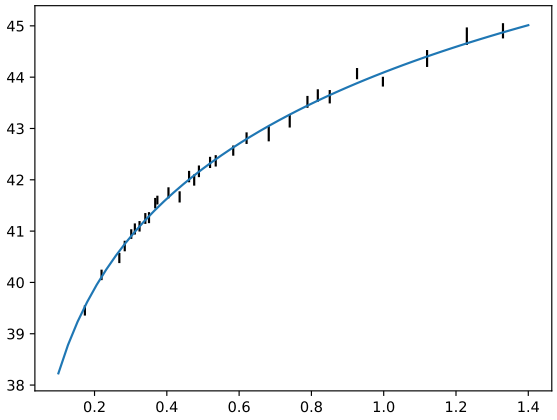
In [2]:
```
# Modulus in different cosmological models
from astropy.cosmology import LambdaCDM

plt.errorbar(z, muB, yerr=dmB, color='k', ls='None')

Om = [0.3, 0.3, 1.0]
OL = [0.7, 0.0, 0.0]
z0 = np.linspace(0.1, 1.4, 50)
label = r'$\Omega_{{\rm m}}={0:g}, \Omega_\Lambda={1:g}$'

for i in range(len(Om)):
    cosmo = LambdaCDM(H0=70, Om0=Om[i], Ode0=OL[i])
    mu = cosmo.distmod(z0)
    plt.plot(z0, mu, label=label.format(Om[i],OL[i]))

plt.xlabel(r'$z$')
plt.ylabel(r'$\mu$')
plt.legend()
plt.show()
```



In [3]:
```
# Fitting of cosmological parameters
from scipy.optimize import minimize
```

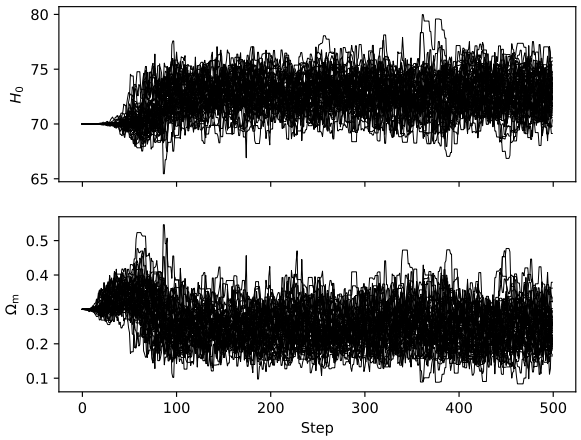
```
        def chi2(par, z, muB, dmB):
            H0, Om, OL = par
            cosmo = LambdaCDM(H0, Om0=Om, Ode0=OL)
            mu = cosmo.distmod(z).value
            diff = ((muB - mu) / dmB)**2
            return diff.sum()

        ini = [70., 0.3, 0.7]
        res = minimize(chi2, ini, args=(z,muB,dmB))

        H0, Om, OL = res.x
        print('Best-fit:')
        print('  H0 = {0:g}'.format(H0))
        print('  Om = {0:g}'.format(Om))
        print('  OL = {0:g}'.format(OL))

        cosmo = LambdaCDM(H0, Om0=Om, Ode0=OL)
        mu = cosmo.distmod(z0)

        plt.errorbar(z, muB, yerr=dmB, color='k', ls='None')
        plt.plot(z0, mu)

        plt.show()
```

```
Best-fit:
  H0 = 73.2867
  Om = 0.244789
  OL = 0.789927
```



In [4]:
```
# MCMC sampling of parameters
from astropy.cosmology import FlatLambdaCDM
import emcee

def chi2_flat(par, z, muB, dmB):
    H0, Om = par
    cosmo = FlatLambdaCDM(H0, Om0=Om)
    mu = cosmo.distmod(z).value
    diff = ((muB - mu) / dmB)**2
    return diff.sum()

def lnprior(par):
    H0, Om = par
    if 50 <= H0 <= 100 and 0 <= Om <= 1:
        return 0.0
    return -np.inf

def lnprob(par, z, muB, dmB):
    lp = lnprior(par)
    if not np.isfinite(lp):
        return -np.inf
    return lp - 0.5 * chi2_flat(par, z, muB, dmB)

ndim, nwalkers = 2, 50
```

```
ini = np.array([70., 0.3])
ini = [ini + 1e-4 * np.random.randn(ndim) \
        for i in range(nwalkers)]
sampler = emcee.EnsembleSampler(nwalkers, ndim, \
        lnprob, args=(z, muB, dmB))
sampler.run_mcmc(ini, 500)

ax = [None] * ndim
for i in range(ndim):
    ax[i] = plt.subplot2grid((ndim,1), (i,0))
    if i != ndim - 1:
        ax[i].set_xticklabels(())

for i in range(nwalkers):
    ax[0].plot(sampler.chain[i, :, 0], 'k-', lw=0.5)
    ax[1].plot(sampler.chain[i, :, 1], 'k-', lw=0.5)

ax[0].set_ylabel(r'$H_0$')
ax[1].set_ylabel(r'$\Omega_{\rm m}$')
ax[1].set_xlabel('Step')
plt.show()
```




```
In [5]: # Likelihood distribution of parameters
        import corner
```

```
from IPython.display import display, Math

samples = sampler.chain[:, 100:, :].reshape((-1, ndim))
fig = corner.corner(samples, labels=[r'$H_0$', \
        r'$\Omega_{\rm m}$', r'$\Omega_\Lambda$'])
fig.show()

H0, Om = map(lambda v: (v[1], v[2]-v[1], v[1]-v[0]), \
        zip(*np.percentile(samples, [16, 50, 84], axis=0)))

display(Math(r'$H_0 = {0:.1f}_{{-{1:.2f}}}^{{+{2:.2f}}}$' \
        ''.format(H0[0], H0[1], H0[2])))
display(Math(r'$\Omega_{{\rm m}} = {0:.2f}_{{-{1:.3f}}}^' \
        '{{+{2:.3f}}}$'.format(Om[0], Om[1], Om[2])))
```

$H_0 = 72.8^{+1.65}_{-1.62}$
$\Omega_\mathrm{m} = 0.24^{+0.051}_{-0.058}$