Hz_code

June 3, 2019

```
In [1]: %matplotlib inline
        import numpy as np
        import matplotlib.pyplot as plt
        import scipy
        import pandas as pd
        import lmfit
        import astropy
        import astropy.units as u
        import corner
        from astropy.cosmology import Flatw0waCDM
```

```
In [2]: # Astropy.cosmology version.Here I am using FlatwowaCDM because it has less parameters
        def Hz1(z1,H01,Om1,w01,wa1):
            cosmo =Flatw0waCDM(H0=H01* u.km / u.s / u.Mpc, Om0=Om1,w0=w01,wa=wa1)
            #the redshift dependence of the dark energy density:
            I = cosmo.de_density_scale(z1)
            E = np.sqrt((Om1*(1+z1)**3.) + (1.-Om1)*I)
            HZ1 = H01*E
            return HZ1
```

```
In [3]: #Define  random redshifts
        num_zs = 1000
        zs= np.linspace(0.01, 3., num=num_zs)
```

```
In [4]: #From Planck observations for CPL mode
        O_m= 0.3029
        w_0= -0.9414
        w_1= -0.4303
        H_0= 68.5265
```

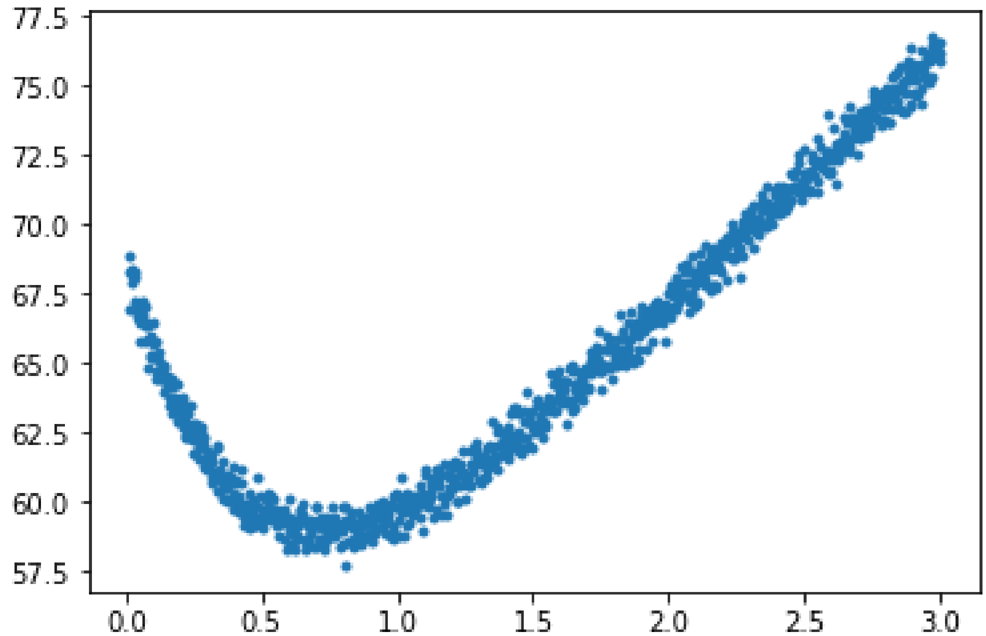
```
In [5]: #Hubble comoving parameter
        Hz=Hz1(zs,H_0,O_m,w_0,w_1)/(1+zs)
```

```
In [6]: #random error for Hz
        error_sigma = 0.5
        e1 = np.random.normal(0., error_sigma, Hz.shape)
```

```
In [7]: H_zz= Hz + e1
```

In [8]:
```
plt.plot(zs,H_zz, '.')
```

Out[8]:
```
[<matplotlib.lines.Line2D at 0x7f0fc55d94d0>]
```



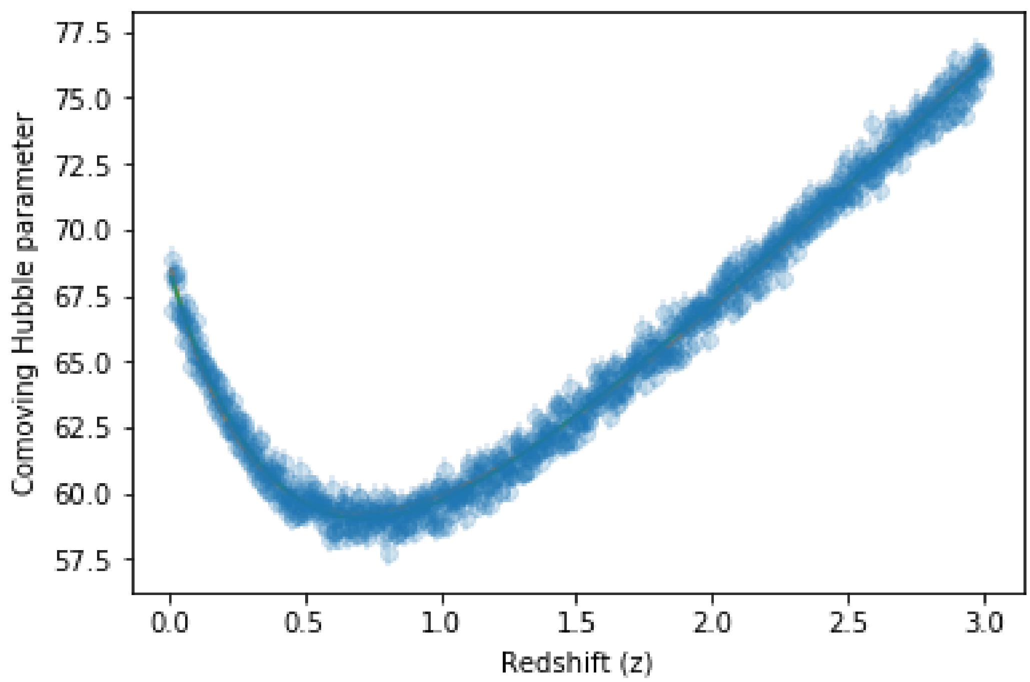
In [9]:
```
p = lmfit.Parameters()
p.add_many(('Om1',0.3,True,0.,2.),
           ('H01',70.0,True,50.,100.),
           ('w01',-1.0,True,-2.,2.),
           ('wa1',-0.1,True,-1.,1.))

def residual(p):
    v = p.valuesdict()
    return (Hz1(zs,v['H01'],v['Om1'],v['w01'],v['wa1'])-H_zz)/error_sigma
```

In [10]:
```
mi = lmfit.minimize(residual, p, method='leastsq', nan_policy='omit')
```

In [11]:
```
plt.errorbar(zs,H_zz,yerr=error_sigma,fmt='o',alpha=0.2)
bestOm1 = mi.params.valuesdict()['Om1']
bestH01 = mi.params.valuesdict()['H01']
bestw01 = mi.params.valuesdict()['w01']
bestwa1 = mi.params.valuesdict()['wa1']
plt.plot(zs,Hz1(zs,bestH01,bestOm1,bestw01,bestwa1))
plt.plot(zs,Hz)
plt.xlabel('Redshift (z)')
plt.ylabel('Comoving Hubble parameter')
lmfit.report_fit(mi)
```

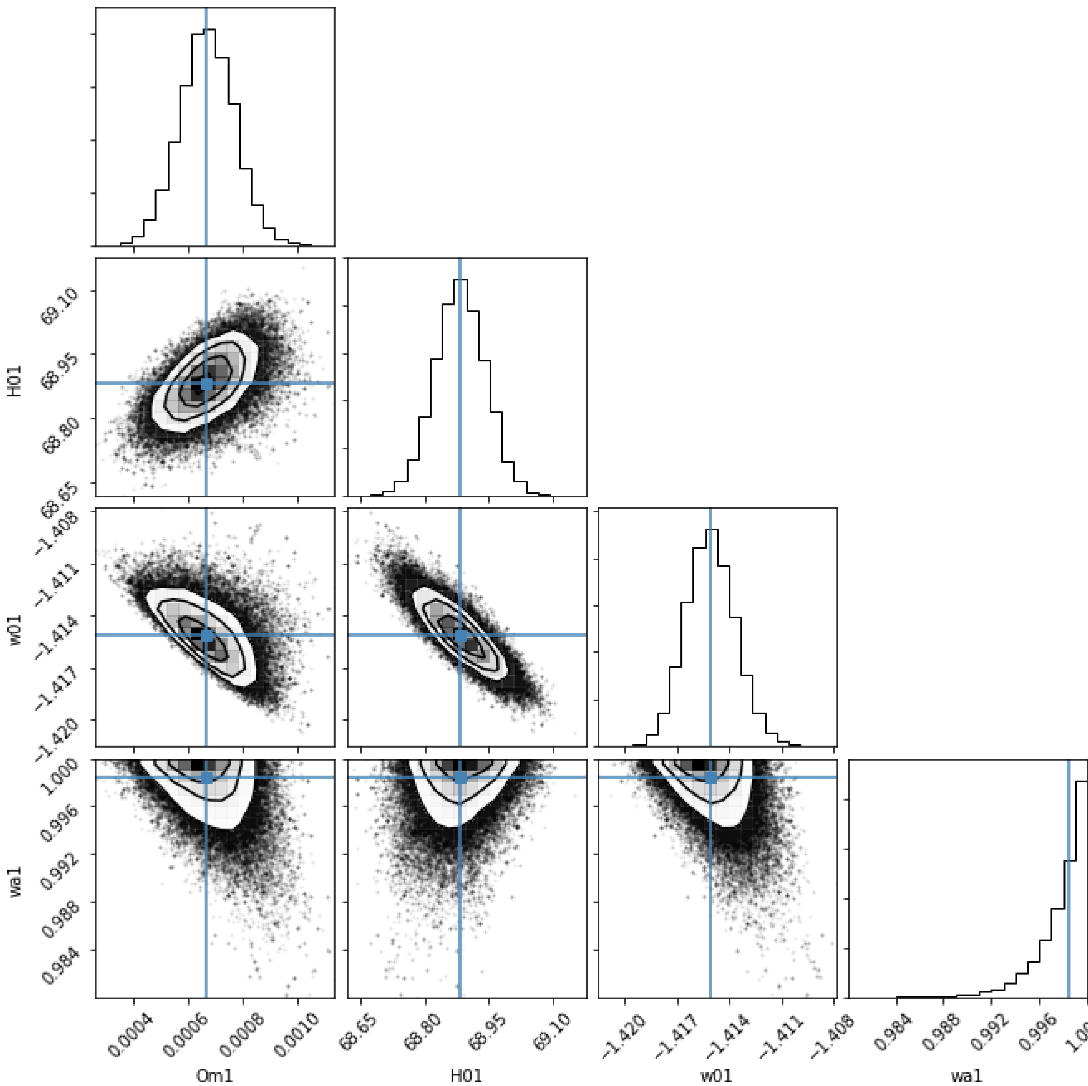
```
[[Fit Statistics]]
    # fitting method   = leastsq
    # function evals   = 63
    # data points      = 1000
    # variables        = 4
    chi-square         = 1054.64781
    reduced chi-square = 1.05888334
    Akaike info crit   = 61.2068816
    Bayesian info crit = 80.8379027
[[Variables]]
    Om1:  6.2587e-04 +/- 3.3977e-04 (54.29%) (init = 0.3)
    H01:  68.8925464 +/- 0.09522049 (0.14%) (init = 70)
    w01: -1.41561150 +/- 0.00467824 (0.33%) (init = -1)
    wa1:  1.00000000 +/- 0.00109956 (0.11%) (init = -0.1)
[[Correlations]] (unreported correlations are < 0.100)
    C(Om1, wa1) = -0.959
    C(w01, wa1) = -0.957
    C(H01, w01) = -0.898
    C(Om1, w01) =  0.839
    C(H01, wa1) =  0.770
    C(Om1, H01) = -0.613
```

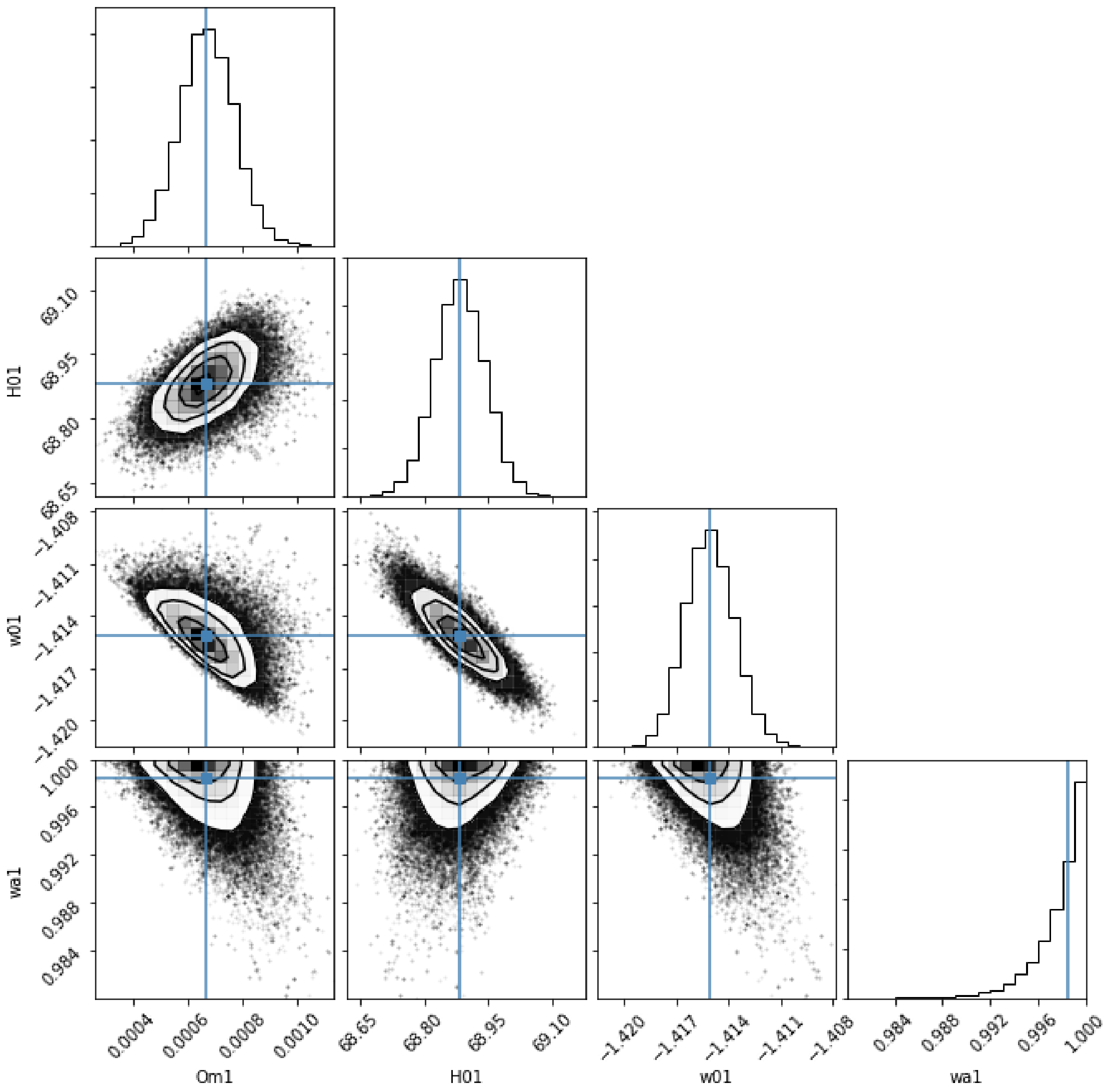



```
In [14]: res = lmfit.minimize(residual, method='emcee', nan_policy='omit', burn=300, steps=150
                                  params=p, is_weighted=True)
```

```
In [15]: corner.corner(res.flatchain,labels=res.var_names, truths=list(res.params.valuesdict()
```

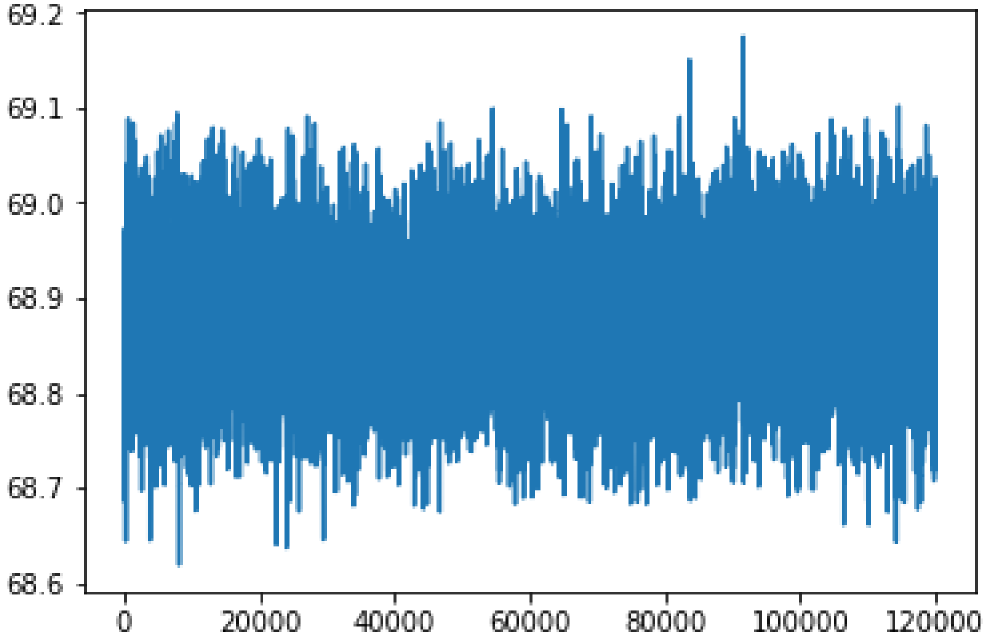
Out[15]:






```
In [16]: plt.plot(res.flatchain.H01)

Out[16]: [<matplotlib.lines.Line2D at 0x7f0fc0f70090>]
```



69.2
69.1
69.0
68.9
68.8
68.7
68.6
0
20000
40000
60000
80000
100000
120000

In [ ]: