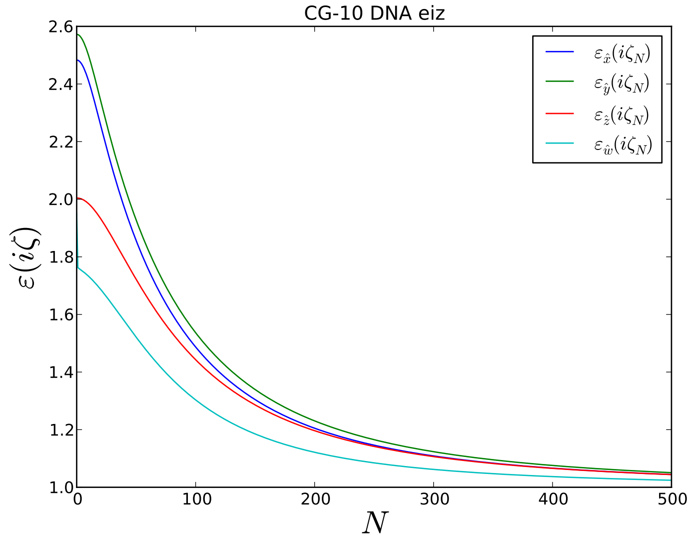
Free energy between two parallel cylinders (CG-10 in water). Nonretarded result, function of separation ↕
Equation 31: $G(\ell,\theta) = -\frac{(\pi R_1^2)(\pi R_2^2)}{2\pi\ \ell^4 \sin\theta}\left(\mathscr{A}^{(0)}(\ell) + \mathscr{A}^{(2)}(\ell)\cos 2\theta\right)$
/usr/bin/python

```python
import numpy as np
import scipy.optimize as opt
from scipy.integrate import trapz
import matplotlib.pyplot as pl
import pyreport
from matplotlib import axis as ax
# use pyreport -l file.py
from pylab import show
from matplotlib.ticker import MultipleLocator
from mpl_toolkits.mplot3d import Axes3D
from pylab import pause
```

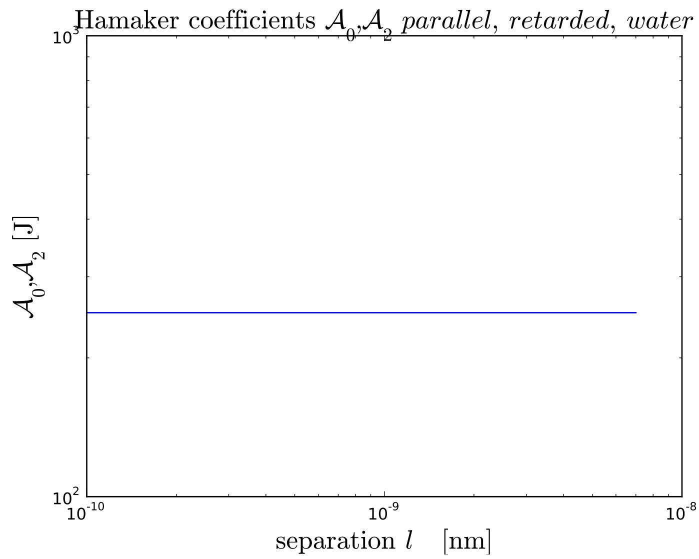
Problem 1) Prove that: $= \frac{1}{N-1}\left(N\sigma^2 - \sigma^2\right)$
Null Hypothesis: the radioactive counts have a Poisson distribution with mean μ.

```python

eiz_x = np.loadtxt('data/eiz_x_output_eV.txt') #perpendicular, radial

eiz_y = np.loadtxt('data/eiz_y_output_eV.txt')
eiz_z = np.loadtxt('data/eiz_z_output_eV.txt') # parallel,axial

eiz_w = np.loadtxt('data/eiz_w_output_eV.txt') # water as intervening medium


#eiz_w[0] = eiz_w[1] #NOTE: there is a jump from first val down to second val

r_1 = 0.5e-9
r_2 = 0.5e-9
c = 2.99e8 # in m/s

# at RT, 1 kT = 4.11e-21 J
T = 297
# h_bar = 1. #1.0546e-34 #in Js
#kb = 8.6173e-5  # in eV/K
kb  = 1.3807e-23 # in J/K

# NOTES:
# z_n_eV = (2*pi*kT/h_bar)n
#        = (0.159 eV) / (6.5821e-16 eVs)
#        = n*2.411e14 rad/s
# z_n_J = (2*pi*kT/h_bar)n
#        = (1.3807e-23 J/K) / (1.0546e-34 Js))*n
#        = n*2.411e14 rad/s
#coeff = 0.159 # in eV w/o 1/h_bar
coeff = 2.411e14 # in rad/s


ns = np.arange(0.,500.)
z = ns * coeff
ls = np.linspace(0.1e-9, 7.0e-9, 10)

def Aiz(perp, par,med):
        return (2.0*(perp-med)*med)/((perp+med)*(par-med))
def ys(a):
        term1 = np.log(3.0 * 5.*(a + a))
        return np.exp(term1)
def y_2s(a):
```

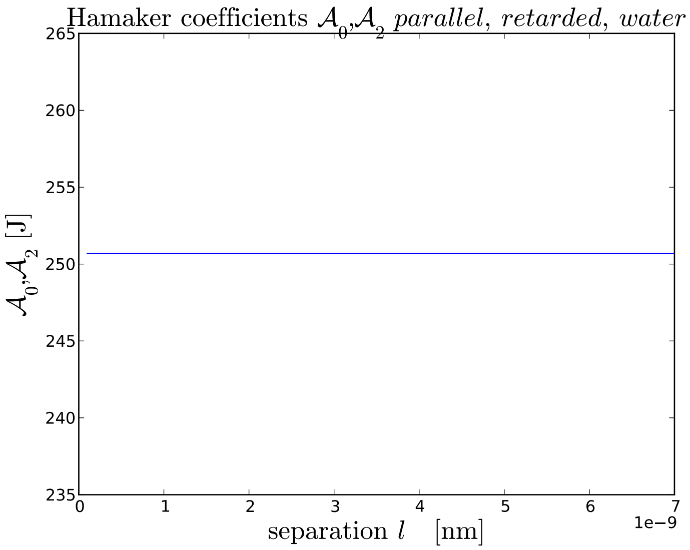
```
        term1 = np.log((19.*a*a))
        return np.exp(term1)
def As(eizz,eizw,Y):
        term1 = ((eizz-eizw)/eizw)*((eizz-eizw)/eizw)
        term2 = Y
        return term1 * term2
def A_2s(eizz,eizw, Y):
        term1 = ((eizz-eizw)/eizw)*((eizz-eizw)/eizw)
        term2 = Y
        return term1 * term2

A   = np.zeros(shape=(len(ns),len(ls)))
A_2 = np.zeros(shape=(len(ns),len(ls)))
aiz = []
sum_A = np.empty(len(ls))
sum_A_2 = np.empty(len(ls))
EL = np.zeros(len(ls))
G_l_t_dt = np.empty(len(ls))

aiz = Aiz(eiz_x,eiz_z, eiz_w) # of length = len(ns)


for k,length in enumerate(ls):
        for j,n in enumerate(ns):
                #print "on n=%d of %d"%(j,len(ns))
                # Integrand:
                y   = ys(aiz[j])
                y_2 = y_2s(aiz[j])
                A[j,k]   = As(eiz_z[j],eiz_w[j],y)
                A_2[j,k] = A_2s(eiz_z[j],eiz_w[j],y_2)
                A[0,k] = (1./2)*A[0,k]
                A_2[0,k] = (1./2)*A_2[0,k]
        sum_A = np.sum(A,axis=0)
        #print 'shape sum_A = ', np.shape(sum_A)
        sum_A_2 = np.sum(A_2,axis=0)
        #print 'shape sum_A_2 = ', np.shape(sum_A_2)
        #sys.exit()
        EL[k] = 1./(length*length*length*length*length)
        G_l_t_dt[k] = kb * T * (9./(32.*64.)) * EL[k]*np.pi*r_1*r_1*r_2*r_2*(
            sum_A[k] + sum_A_2[k])
        #print 'theta = %.1f, length = %.1f, G = %s' %(i,k,G_l_t_dt[k,i])

pl.figure()
pl.plot(ns,eiz_x, color = 'b', label = r'$\varepsilon_{\hat{x}}(i\zeta_{N})$')
pl.plot(ns,eiz_y, color = 'g', label = r'$\varepsilon_{\hat{y}}(i\zeta_{N})$')
pl.plot(ns,eiz_z, color = 'r', label = r'$\varepsilon_{\hat{z}}(i\zeta_{N})$')
pl.plot(ns,eiz_w, color = 'c', label = r'$\varepsilon_{\hat{w}}(i\zeta_{N})$')
pl.xlabel(r'$N$', size = 24)
pl.ylabel(r'$\varepsilon(i\zeta)$', size = 24)
pl.legend()
pl.title(r'CG-10 DNA eiz')
show()
```



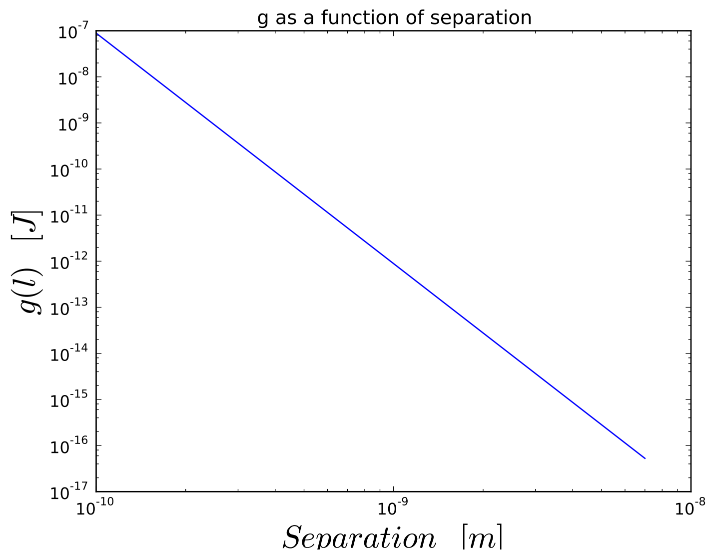

```python

pl.figure()
pl.loglog(ls,sum_A+sum_A_2,'b-')
pl.xlabel(r'$\mathrm{separation}\,\it{l}\,\,\,\rm{[nm]}$', size = 20)
pl.ylabel(r'$\mathrm{\mathcal{A_{0},A_{2}}\,[J]}$', size = 20)
pl.title(r'$\mathrm{Hamaker \, coefficients \,\mathcal{A_{0},A_{2}}}\, parallel
    ,\,retarded,\,water$', size = 20)
show()
```



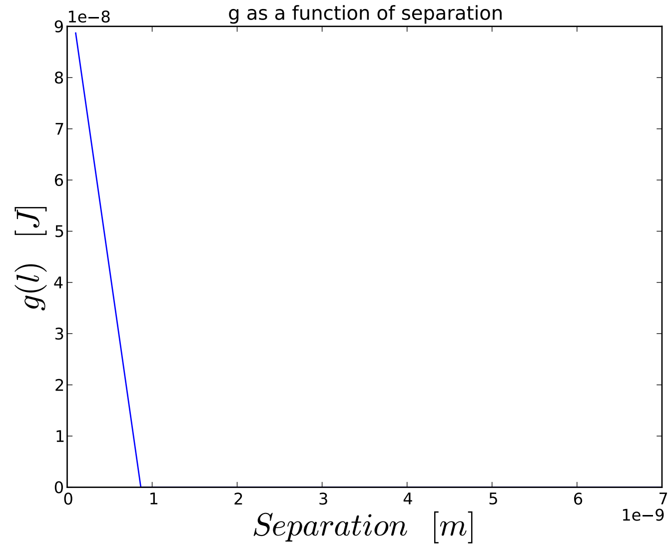

```python

pl.figure()
pl.plot(ls,sum_A+sum_A_2,'b-')
pl.xlabel(r'$\mathrm{separation}\,\it{l}\,\,\,\rm{[nm]}$', size = 20)
pl.ylabel(r'$\mathrm{\mathcal{A_{0},A_{2}}\,[J]}$', size = 20)
pl.title(r'$\mathrm{Hamaker \, coefficients \,\mathcal{A_{0},A_{2}}}\, parallel
    ,\,retarded,\,water$', size = 20)
show()
```




```
pl.figure()
pl.loglog(ls, G_l_t_dt)
pl.xlabel(r'$Separation\,\,[m]$', size = 24)
pl.ylabel(r'$g(l)\,\,[J]$', size = 24)
#pl.axis([1.5e-9, 6.5e-8,100,145])
pl.title('g as a function of separation')
show()
```




```
pl.figure()
pl.plot(ls, G_l_t_dt)
pl.xlabel(r'$Separation\,\,[m]$', size = 24)
pl.ylabel(r'$g(l)\,\,[J]$', size = 24)
#pl.axis([1.5e-9, 6.5e-8,100,145])
pl.title('g as a function of separation')
show()
```


g as a function of separation
1e−8
9
8
7
6
5
4
3
2
1
0
$g(l) \quad [J]$
0
1
2
3
4
5
6
7
1e−9
$Separation \quad [m]$