```
#
# calculates the likelihood for a site at the root
#
# load the scipy and numpy ibrary
import numpy as np
import scipy as sc
from scipy import linalg
# we assume that basefreq and h are numpy arrays
# this functions multiplies each element of basefreq and h and sums up
def siteLikelihood(basefreq, h):
    return np.dot(basefreq,h)

def likelihood(sitelikes):
    return np.sum(np.log(sitelikes))

# conditional likelihood calculates h from g using a probability matrix
# we need an array h
#            array q
#            branchlength t
def condLikelihood(h, q, t):
    p = linalg.expm2(q * t)
    print "Transition probability matrix for branchlength", t
    print p
    return np.dot(h, p)

#
#
# testing
# preparation
basefreq = np.array([0.25,0.25,0.25,0.25])
h1 = np.array([0.005,0.00001,0.005, 0.0076])
h2 = np.array([0.005,0.00001,0.005, 0.0076])
print "Test 1"
s1 = siteLikelihood(basefreq,h1)
s2 = siteLikelihood(basefreq,h2)
print "site likelihoods:", s1,s2
sitelikelihoods = np.array([s1,s2])
print sitelikelihoods
print "total log(likelihood)", likelihood(sitelikelihoods)

####
# test 2
# print the likelihood of a small tree
# ((a:0.3,b:0.3):0.3,c:0.6)
# a   b   c
# +-d-+   |
#   |     |
#   +--e--+
#      |
# Transition rate matrix for JC69
q = np.array(
[[-1,1./3,1./3,1./3],
[1./3,-1,1./3,1./3],
```

```
54 [1./3,1./3,-1.0,1./3],
55 [1./3,1./3,1./3,-1.0]]
56 )
57 ga = np.array((1.0,0.0,0.0,0.0))
58 gb = np.array((1.0,0,0,0))
59 gc = np.array((0.0,1.0,0,0))
60
61 print "Test 2"
62 t= 0.3 # branchlength for a to d and b to d
63 c1 = condLikelihood(ga,q,t)
64 c2 = condLikelihood(gb,q,t)
65 h = c1 * c2
66 t= 0.3 # branchlength for a to d and b to d
67 ha = condLikelihood(ga,q,t)
68 hb = condLikelihood(gb,q,t)
69 gd = ha * hb

70 t = 0.3 # branchlength for d to e
71 hd = condLikelihood(gd,q,t)
72 t = 0.6 # branchlength for c to e
73 hc = condLikelihood(gc,q,t)
74 ge = hd * hc
75 sitelikelihoods=np.array(ge)
76 print "total log(likelihood)", likelihood(sitelikelihoods)
```