Homework #5

- Using the supplied Python program (see p.3), calculate the total log-likelihood and the site log-likelihoods for the following 2 sites under the **JC model** assuming that **all edges have length 0.2**. Note that you can (and should!) use PAUP* to check the total log-likelihood for each step of this homework (create a NEXUS file like that used in the likelihood lab).

\[
L = \Pr(r_1) \Pr(D|r_1) + \Pr(r_2) \Pr(D|r_2)
\]

\[
= p_{\text{invar}} \Pr(D|r_1) + (1 - p_{\text{invar}}) \Pr(D|r_2)
\]

<table>
<thead>
<tr>
<th>Site 1</th>
<th>Site 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Log L =</td>
<td>Log L =</td>
</tr>
<tr>
<td>L1</td>
<td>L2</td>
</tr>
</tbody>
</table>

- Now **modify** the Python program to compute the site log-likelihoods under a **JC+I model** (tree same, \( p_{\text{invar}} = 0.5 \)). The likelihood for each site must take account of the two possible relative rates: \( r_1 = 0.0 \) and \( r_2 = 1/(1 - p_{\text{invar}}) \). Note that the function sitelike does 99% of the work for you and you will **not need to modify it**! Simply pass in the rate you want to use as the last argument to compute \( \Pr(D|r_1) \) and \( \Pr(D|r_2) \) below:

\[
L_i = \Pr(r_1) \Pr(D|r_1) + \Pr(r_2) \Pr(D|r_2)
\]

\[
= p_{\text{invar}} \Pr(D|r_1) + (1 - p_{\text{invar}}) \Pr(D|r_2)
\]

<table>
<thead>
<tr>
<th>Site 1</th>
<th>Site 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Log L =</td>
<td>Log L =</td>
</tr>
<tr>
<td>L1</td>
<td>L2</td>
</tr>
</tbody>
</table>

Is the total log-likelihood higher for JC or JC+I? _________
Now modify the Python program again to compute the site log-likelihoods under a **JC+G model** (tree same, **gamma shape=0.2, 4 categories**). Use PAUP* to obtain the 4 relative rates (issue the command **“gammaplot shape=0.2”**: the relative rates are in the last column.)

\[
\begin{align*}
\log L_1 &= \text{______________} & \log L_2 &= \text{______________} & \text{Total log-likelihood:} \\
&\text{(use 6 decimal places)} & &\text{(use 6 decimal places)} & \text{(use 6 decimal places)}
\end{align*}
\]

Which model has the highest total log-likelihood: JC, JC+I, or JC+G? __________

Use PAUP* to answer these questions (keep the edge lengths fixed at 0.2 but estimate pinvar for the JC+I model and shape for the JC+G model):

What is the MLE for pinvar in the JC+I model? __________ max log L? __________

What is the MLE for shape in the JC+G model? __________ max log L? __________

Which model fits best when the likelihood is maximized: JC, JC+I, or JC+G? __________

Email me (paul.lewis@uconn.edu) your modified Python program(s). Ideally, the output of your program(s) should clearly show the two site log-likelihoods and the model under consideration. Ideally, here is what your program should output:

**JC+I model (brlens = 0.2, pinvar = 0.5):**
- log-likelihood for site 1 = -1.93115539911
- log-likelihood for site 2 = -7.4688159219
- total log-likelihood = -9.39997132102

**JC+G model (brlens = 0.2, shape = 0.2):**
- log-likelihood for site 1 = -1.77778297625
- log-likelihood for site 2 = -7.36834049814
- total log-likelihood = -9.14612347438
from math import exp, log

# You should *not* modify this function
# same_list holds number of edges with the same state at both ends
# v is the edge length
# r is the relative rate
def sitelike(same_list, v, r):
    probsame = 0.25 + 0.75*exp(-4.0*r*v/3.0)
    probdiff = 0.25 - 0.25*exp(-4.0*r*v/3.0)
    like = 0.0
    for n in same_list:
        nsame=float(n)
        ndiff=float(5-n)
        like += 0.25*(probsame**nsame)*(probdiff**ndiff)
    return like

same1 = [5, 2, 2, 2, 2, 1, 0, 0, 2, 0, 1, 0, 2, 0, 0, 1]
same2 = [2, 1, 2, 2, 1, 2, 2, 2, 0, 0, 2, 1, 0, 0, 1, 2]

# Results for JC model
like1 = sitelike(same1, 0.2, 1.0)
like2 = sitelike(same2, 0.2, 1.0)
print 'JC model (brlens = 0.2):'
print '  log-likelihood for site 1 =', log(like1)
print '  log-likelihood for site 2 =', log(like2)
print '  total log-likelihood =', log(like1)+log(like2)

# Results for JC+I model
# (add your Python code here)

# Results for JC+G model
# (add your Python code here)

Note: you may lose the indentation when you copy and paste: be sure to indent your python code correctly before running!