PAML (Phylogenetic Analysis by Maximum Likelihood)

A program package by Ziheng Yang

(Demonstration by Joseph Bielawski)

What does PAML do?

Features include:

- estimating synonymous and nonsynonymous rates
- testing hypotheses concerning d_N/d_S rate ratios
- various amino acid-based likelihood analysis
- ancestral sequence reconstruction (DNA, codon, or AAs)
- various clock models
- simulating nucleotide, codon, or AA sequence data sets
- and more ……

Programs in the package

baseml	for bases
basemlg	continuous-gamma for bases
codeml	aaml (for amino acids) & codonml (for codons)
evolver	simulation, tree distances
yn00	d_N and d_S by YN00
chi2	chi square table
pamp	parsimony (Yang and Kumar 1996)
mcmctree	Bayes MCMC tree (Yang & Rannala 1997). Slow

Downloading PAML

PAML download files are at:

ftp://abacus.gene.ucl.ac.uk/pub/paml/

For windows, Macs, and Unix/Linux

Running PAML programs

1. Sequence data file
2. Tree file
3. Control file (*.ctl)

Running PAML programs: the sequence file

```
4 20
sequence_1  TCATT CTATC TATCG TGATG
sequence_2  TCATT CTATC TATCG TGATG
sequence_3  TCATT CTATC TATCG TGATG
sequence_4  TCATT CTATC TATCG TGATG
```

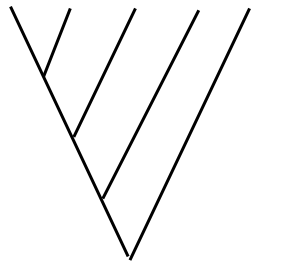
```
4 20
sequence_1TCATTCTATCTATCGTGATG
sequence_2TCATTCTATCTATCGTGATG
sequence_3TCATTCTATCTATCGTGATG
sequence_4TCATTCTATCTATCGTGATG
```

Format = plain text in “PHYLIP” format

Running PAML programs: the tree file

Format = parenthetical notation

```
((((1 , 2), 3), 4), 5)
```

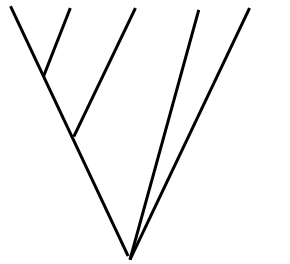


This is a rooted tree (root is degree = 2)

Running PAML programs: the tree file

Format = parenthetical notation

```
(((1 , 2), 3), 4, 5)
```



This is an **unrooted** tree (basal node is degree = 3)

Running PAML programs: the tree file

Format = parenthetical notation

Examples:

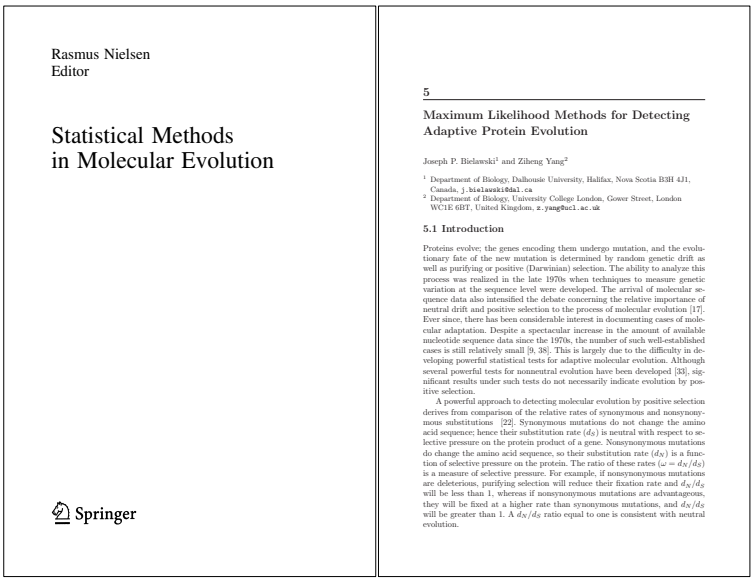
```
(((1,2),3),4,5);

((((1,2),3),4),5);

(((1:0.1, 2:0.2):0.8, 3:0.3):0.7, 4:0.4, 5:0.5);

(((Human:0.1, Chimpanzee:0.2):0.8, Gorilla:0.3):0.7,
Orangutan:0.4, Gibbon:0.5);
```


Running PAML programs: the “*.ctl” file

codeml.ctl

```
  seqfile = seqfile.txt   * sequence data filename
 treefile = tree.txt      * tree structure file name
  outfile = results.txt   * main result file name

    noisy = 9      * 0,1,2,3,9: how much rubbish on the screen
  verbose = 1      * 1:detailed output
  runmode = 0      * 0:user defined tree

  seqtype = 1      * 1:codons
CodonFreq = 2      * 0:equal, 1:F1X4, 2:F3X4, 3:F61

    model = 0      * 0:one omega ratio for all branches

  NSsites = 0      * 0:one omega ratio (M0 in Tables 2 and 4)
                   * 1:neutral (M1 in Tables 2 and 4)
                   * 2:selection (M2 in Tables 2 and 4)
                   * 3:discrete (M3 in Tables 2 and 4)
                   * 7:beta (M7 in Tables 2 and 4)
                   * 8:beta&w; (M8 in Tables 2 and 4)

    icode = 0      * 0:universal code

fix_kappa = 0      * 1:kappa fixed, 0:kappa to be estimated
    kappa = 2      * initial or fixed kappa

fix_omega = 0      * 1:omega fixed, 0:omega to be estimated
    omega = 5      * initial omega

                   *set ncatG for models M3, M7, and M8!!!
   *ncatG = 3      * # of site categories for M3 in Table 4
   *ncatG = 10     * # of site categories for M7 and M8 in Table 4
```


Exercises:



Rasmus Nielsen
Editor

Statistical Methods
in Molecular Evolution

Springer

5

Maximum Likelihood Methods for Detecting Adaptive Protein Evolution

Joseph P. Bielawski[1] and Ziheng Yang[2]

[1] Department of Biology, Dalhousie University, Halifax, Nova Scotia B3H 4J1, Canada, j.bielawski@dal.ca
[2] Department of Biology, University College London, Gower Street, London WC1E 6BT, United Kingdom, z.yang@ucl.ac.uk

5.1 Introduction

Proteins evolve; the genes encoding them undergo mutation, and the evolutionary fate of the new mutation is determined by random genetic drift as well as purifying or positive (Darwinian) selection. The ability to analyze this process was realized in the late 1970s when techniques to measure genetic variation at the sequence level were developed. The arrival of molecular sequence data also intensified the debate concerning the relative importance of neutral drift and positive selection to the process of molecular evolution [17]. Ever since, there has been considerable interest in documenting cases of molecular adaptation. Despite a spectacular increase in the amount of available nucleotide sequence data since the 1970s, the number of such well-established cases is still relatively small [9, 38]. This is largely due to the difficulty in developing powerful statistical tests for adaptive molecular evolution. Although several powerful tests for nonneutral evolution have been developed [33], significant results under such tests do not necessarily indicate evolution by positive selection.

A powerful approach to detecting molecular evolution by positive selection derives from comparison of the relative rates of synonymous and nonsynonymous substitutions [22]. Synonymous mutations do not change the amino acid sequence; hence their substitution rate (d_S) is neutral with respect to selective pressure on the protein product of a gene. Nonsynonymous mutations do change the amino acid sequence, so their substitution rate (d_N) is a function of selective pressure on the protein. The ratio of these rates ($\omega = d_N/d_S$) is a measure of selective pressure. For example, if nonsynonymous mutations are deleterious, purifying selection will reduce their fixation rate and d_N/d_S will be less than 1, whereas if nonsynonymous mutations are advantageous, they will be fixed at a higher rate than synonymous mutations, and d_N/d_S will be greater than 1. A d_N/d_S ratio equal to one is consistent with neutral evolution.

Exercises:

	Method/model	program	dataset
1	Pair-wise ML method	codeml	*Drosophila GstD1*
2	Pair-wise ML method	codeml	*Drosophila GstD1*
3	M0 and "branch models"	codeml	*Ldh* gene family
4	M0 and "site models"	codeml	HIV-2 *nef* genes

Exercise 1: **ML estimation of the d_N/d_S (ω) ratio "by hand" for *GstD1***

Dataset: *GstD1* genes of *Drosophila melanogaster* and *D. simulans* (600 codons).

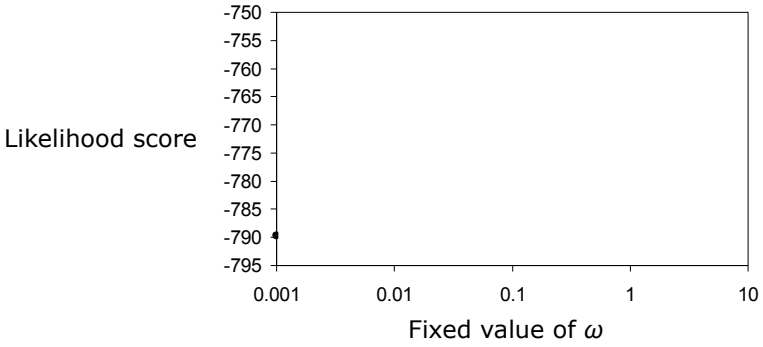
Objective: Use codeml to evaluate the likelihood the *GstD1* sequences for a variety of fixed ω values.

1- Plot log-likelihood scores against the values of ω and determine the maximum likelihood estimate of ω.

2- Check your finding by running codeml's hill-climbing algorithm.

```
  seqfile = seqfile.txt   * sequence data filename
  outfile = results.txt   * main result file name

    noisy = 9      * 0,1,2,3,9: how much rubbish on the screen
  verbose = 1      * 1:detailed output
  runmode = -2     * -2:pairwise

  seqtype = 1      * 1:codons
CodonFreq = 3      * 0:equal, 1:F1X4, 2:F3X4, 3:F61
    model = 0      *
  NSsites = 0      *
    icode = 0      * 0:universal code

fix_kappa = 0      * 1:kappa fixed, 0:kappa to be estimated
    kappa = 2      * initial or fixed kappa

fix_omega = 1      * 1:omega fixed, 0:omega to be estimated
    omega = 0.001  * 1st fixed omega value  [CHANGE THIS]

   *alternate fixed omega values
   *omega = 0.005  * 2nd fixed value
   *omega = 0.01   * 3rd fixed value
   *omega = 0.05   * 4th fixed value
   *omega = 0.10   * 5th fixed value
   *omega = 0.20   * 6th fixed value
   *omega = 0.40   * 7th fixed value
   *omega = 0.80   * 8th fixed value
   *omega = 1.60   * 9th fixed value
   *omega = 2.00   * 10th fixed value
```


Plot results: likelihood score vs. omega (log scale)



Exercise 1

If you forget what to do, there is a "step-by-step" guide on the course web site.

There is also a "HelpFile" to help you to get what you need from the output of codeml

Exercise 2:	**Investigating the sensitivity of the d_N/d_S ratio to assumptions**
Dataset:	*GstD1* genes of *Drosophila melanogaster* and *D. simulans* (600 codons).
Objective:	1- Test effect of transition / transversion ratio (κ) 2- Test effect of codon frequencies (π_i's) 3- Determine which assumptions yield the largest and smallest values of *S*, and what is the effect on ω

"`CodonFreq=`" is used to specify the equilibrium codon frequencies

Fequal: - 1/61 each for the standard genetic code
- `CodonFreq = 0`
- number of parameters in the model = 0

F3x4: - calculated from the average nucleotide frequencies at the three codon positions
- `CodonFreq = 2`
- number of parameters in the model = 9

F61 - also called "ftable"; empirical estimate of each codon frequency
- `CodonFreq = 3`
- number of parameters in the model = 61

Exercise 2

Part 2: Real data exercises

Example: A → C

AAA → CAA

AAA → ACA

AAA → AAC

	Target codon (nucleotide)			
	CAA	**ACA**	**AAC**	**NP**
No bias	1/61	1/61	1/61	0
F1×4	$\pi_C(\pi_A)^2$	$\pi_C(\pi_A)^2$	$\pi_C(\pi_A)^2$	3
MG	π_C^1	π_C^2	π_C^3	9
F3×4 (GY)	$\pi_C^1\pi_A^2\pi_A^3$	$\pi_A^1\pi_C^2\pi_A^3$	$\pi_A^1\pi_A^2\pi_C^3$	9
F61 (GY)	π_{CAA}	π_{ACA}	π_{AAC}	61

```
      seqfile = seqfile.txt   * sequence data filename
      outfile = results.txt   * main result file name

        noisy = 9      * 0,1,2,3,9: how much rubbish on the screen
      verbose = 1      * 1:detailed output
      runmode = -2     * -2:pairwise

      seqtype = 1      * 1:codons
    CodonFreq = 0      * 0:equal, 1:F1X4, 2:F3X4, 3:F61 [CHANGE THIS]
        model = 0      *
      NSsites = 0      *
        icode = 0      * 0:universal code

    fix_kappa = 1      * 1:kappa fixed, 0:kappa to be estimated [CHANGE THIS]
        kappa = 1      * fixed or initial value

    fix_omega = 0      * 1:omega fixed, 0:omega to be estimated
        omega = 0.5    * initial omega value

* Codon bias = none (equal); Ts/Tv bias = none (fixed at 1)
* Codon bias = none (equal); Ts/Tv bias = Yes (estimate by ML)

* Codon bias = yes (F3x4); Ts/Tv bias = none (fixed at 1)
* Codon bias = yes (F3x4); Ts/Tv bias = Yes (estimate by ML)

* Codon bias = yes (F61); Ts/Tv bias = none (fixed at 1)
* Codon bias = yes (F61); Ts/Tv bias = Yes (estimate by ML)
```


Further details for about the assumptions tested in ACTIVITY 2

Assumption set 1: (Codon bias = none; Ts/Tv bias = none)
CodonFreq=0; kappa=1; fix_kappa=1

Assumption set 2: (Codon bias = none; Ts/Tv bias = Yes)
CodonFreq=0; kappa=1; fix_kappa=0

Assumption set 3: (Codon bias = yes [F3x4]; Ts/Tv bias = none)
CodonFreq=2; kappa=1; fix_kappa=1

Assumption set 4: (Codon bias = yes [F3x4]; Ts/Tv bias = Yes)
CodonFreq=2; kappa=1; fix_kappa=0

Assumption set 5: (Codon bias = yes [F61]; Ts/Tv bias = none)
CodonFreq=3; kappa=1; fix_kappa=1

Assumption set 6: (Codon bias = yes [F61]; Ts/Tv bias = Yes)
CodonFreq=3; kappa=1; fix_kappa=0

Complete this table (If you forget what to do, there is a "step-by-step" guide on the course web-site.)

Table E2: Estimation of d_S and d_N between *Drosophila melanogaster* and *D. simulans GstD1* genes

Assumptions	κ	S	*N*	d_S	d_N	ω	ℓ
Fequal + κ = 1	1.0	?	?	?	?	?	?
Fequal + κ = estimated	?	?	?	?	?	?	?
F3×4 + κ = 1	1.0	?	?	?	?	?	?
F3×4 + κ = estimated	?	?	?	?	?	?	?
F61 + κ = 1	1.0	?	?	?	?	?	?
F61 + κ = estimated	?	?	?	?	?	?	?

κ = transition/transversion rate ratio

S = number of synonymous sites

N = number of nonsynonymous sites

$\omega = d_N/d_S$

ℓ = log likelihood score

Exercise 3: **Test hypotheses about molecular evolution of *Ldh***

Dataset: The *Ldh* gene family is an important model system for molecular evolution of isozyme multigene families. The rate of evolution is known to have increased in *Ldh*-C following the gene duplication event

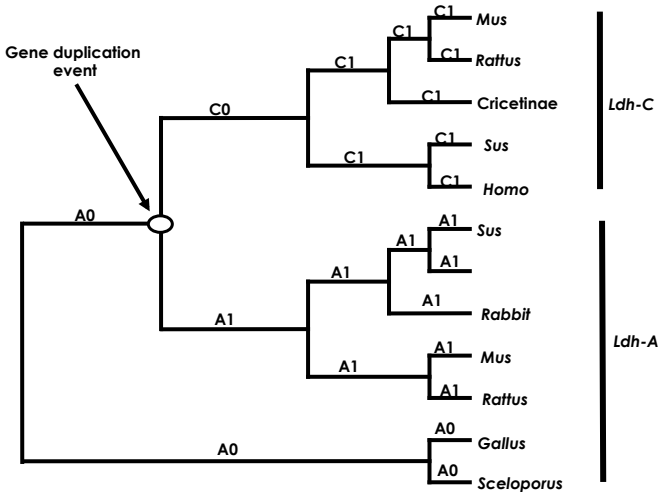
Objective: Use LRTs to evaluate the following hypotheses:

1- The mutation rate of *Ldh-C* has increased relative to *Ldh-A*,

2- A burst of positive selection for functional divergence occurred following the duplication event that gave rise to *Ldh-C*

3- There was a long term shift in selective constraints following the duplication event that gave rise to *Ldh-C*

Exercise 3



H_0: $\omega_{A0} = \omega_{A1} = \omega_{C1} = \omega_{C0}$
H_1: $\omega_{A0} = \omega_{A1} = \omega_{C1} \neq \omega_{C0}$
H_2: $\omega_{A0} = \omega_{A1} \neq \omega_{C1} = \omega_{C0}$
H_3: $\omega_{A0} \neq \omega_{A1} \neq \omega_{C1} = \omega_{C0}$

Exercise 3


```
   seqfile = seqfile.txt   * sequence data filename
  treefile = tree.txt      * tree structure file name [CHANGE THIS]
   outfile = results.txt   * main result file name

     noisy = 9      * 0,1,2,3,9: how much rubbish on the screen
   verbose = 1      * 1:detailed output
   runmode = 0      * 0:user defined tree

   seqtype = 1      * 1:codons
 CodonFreq = 2      * 0:equal, 1:F1X4, 2:F3X4, 3:F61

     model = 0      * 0:one omega ratio for all branches
                    * 1:separate omega for each branch
                    * 2:user specified dN/dS ratios for branches

   NSsites = 0      *

     icode = 0      * 0:universal code

 fix_kappa = 0      * 1:kappa fixed, 0:kappa to be estimated
     kappa = 2      * initial or fixed kappa

 fix_omega = 0      * 1:omega fixed, 0:omega to be estimated
     omega = 0.2    * initial omega
```

```
*H0 in Table 3:
*model = 0
*(X02152Hom,U07178Sus,(M22585rab,((NM017025Rat,U13687Mus),
*(((AF070995C,(X04752Mus,U07177Rat)),(U95378Sus,U13680Hom)),(X53828OG1,
* U28410OG2)))));

*H1 in Table 3:
*model = 2
*(X02152Hom,U07178Sus,(M22585rab,((NM017025Rat,U13687Mus),(((AF070995C,
*(X04752Mus,U07177Rat)),(U95378Sus,U13680Hom))#1,(X53828OG1,U28410OG2))
* )));

*H2 in Table 3:
*model = 2
*  (X02152Hom,U07178Sus,(M22585rab,((NM017025Rat,U13687Mus),(((AF070995C
*  #1,(X04752Mus #1,U07177Rat #1)#1)#1,(U95378Sus #1,U13680Hom #1)
*  #1)#1,(X53828OG1,U28410OG2)))));

*H3 in Table 3:
*model = 2
*  (X02152Hom,U07178Sus,(M22585rab,((NM017025Rat,U13687Mus),(((AF070995C
*  #1,(X04752Mus #1,U07177Rat #1)#1)#1,(U95378Sus #1,U13680Hom #1)
*  #1)#1,(X53828OG1 #2,U28410OG2 #2)#2))));
```

Complete this table (If you forget what to do, there is a "step-by-step" guide on the course web-site.)

Table E3: Parameter estimates under models of variable ω ratios among lineages and LRTs of their fit to the *Ldh-A* and *Ldh-C* gene family.

Models	ω_{A0}	ω_{A1}	ω_{C1}	ω_{C0}	ℓ	LRT
H_0: $\omega_{A0} = \omega_{A1} = \omega_{C1} = \omega_{C0}$	?	= $\omega_{A.0}$	= $\omega_{A.0}$	= $\omega_{A.0}$	?	?
H_1: $\omega_{A0} = \omega_{A1} = \omega_{C1} \neq \omega_{C0}$	?	= $\omega_{A.0}$	= $\omega_{A.0}$	?	?	?
H_2: $\omega_{A0} = \omega_{A1} \neq \omega_{C1} = \omega_{C0}$	?	= $\omega_{A.0}$	?	= $\omega_{C.1}$	?	**?**
H_3: $\omega_{A0} \neq \omega_{A1} \neq \omega_{C1} = \omega_{C0}$	?	?	?	= $\omega_{C.1}$	?	?

The topology and branch specific ω ratios are presented in Figure 5.

H_0 v H_1: df = 1

H_0 v H_2: df = 1

H_2 v H_3: df = 1

$\chi^2_{df=1,\ \alpha=0.05} = 3.841$

Exercise 4: **Testing for adaptive evolution in the *nef* gene of human HIV-2 (Start tonight, but finish as homework)**

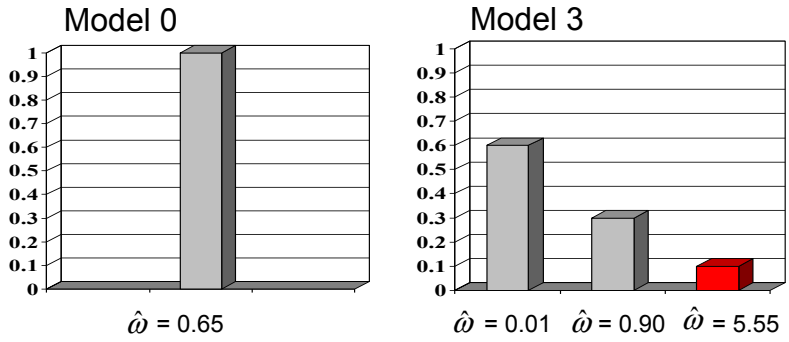
Dataset: 44 *nef* sequences from a study population of 37 HIV-2 infected people living in Lisbon, Portugal. The *nef* gene in HIV-2 has received less attention than HIV-1, presumably because HIV-2 is associated with reduced virulence and pathogenicity relative to HIV-1

Objectives: 1- Learn to use LRTs to test for sites evolving under positive selection in the *nef* gene.

2- If you find significant evidence for positive selection, then identify the involved sites by using empirical Bayes methods.

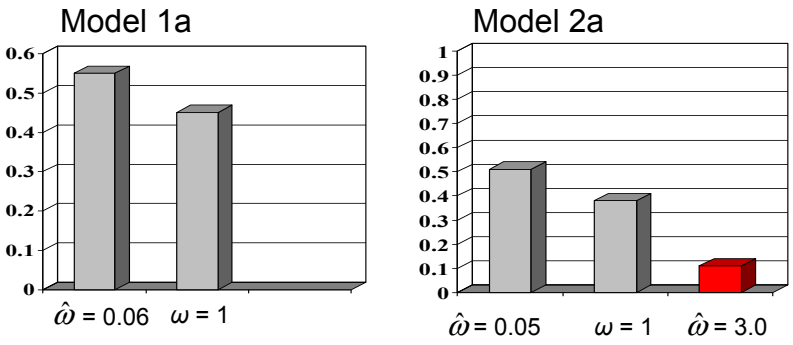
H_0: uniform selective pressure among sites (M0)
H_1: variable selective pressure among sites (M3)

Compare $2\Delta l = 2(l_1 - l_0)$ with a χ^2 distribution



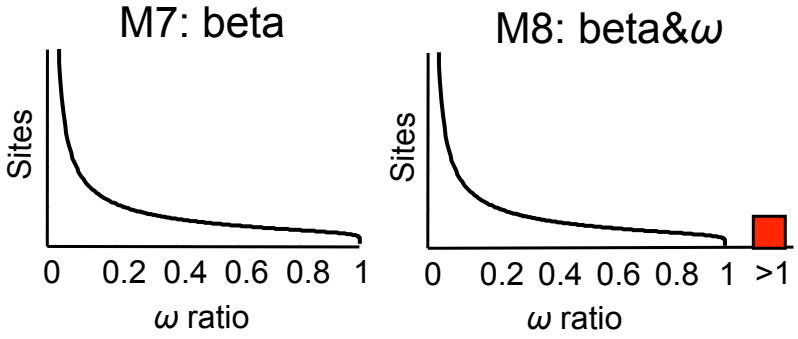
H_0: variable selective pressure but NO positive selection (M1)
H_1: variable selective pressure with positive selection (M2)

Compare $2\Delta l = 2(l_1 - l_0)$ with a χ^2 distribution



H_0: Beta distributed variable selective pressure (M7)
H_1: Beta plus positive selection (M8)

Compare $2\Delta l = 2(l_1 - l_0)$ with a χ^2 distribution




```
      seqfile = seqfile.txt          * sequence data filename

  * treefile = treefile_M0.txt       * SET THIS for tree file with ML branch lengths under M0
  * treefile = treefile_M1.txt       * SET THIS for tree file with ML branch lengths under M1
  * treefile = treefile_M2.txt       * SET THIS for tree file with ML branch lengths under M2
  * treefile = treefile_M3.txt       * SET THIS for tree file with ML branch lengths under M3
  * treefile = treefile_M7.txt       * SET THIS for tree file with ML branch lengths under M7
  * treefile = treefile_M8.txt       * SET THIS for tree file with ML branch lengths under M8

    outfile = results.txt            * main result file name
      noisy = 9                      * lots of rubbish on the screen
    verbose = 1                      * detailed output
    runmode = 0                      * user defined tree
    seqtype = 1                      * codons
  CodonFreq = 2                      * F3X4 for codon ferquencies
      model = 0                      * one omega ratio for all branches

  * NSsites = 0                      * SET THIS for M0
  * NSsites = 1                      * SET THIS for M1
  * NSsites = 2                      * SET THIS for M2
  * NSsites = 3                      * SET THIS for M3
  * NSsites = 7                      * SET THIS for M7
  * NSsites = 8                      * SET THIS for M8

      icode = 0                      * universal code
  fix_kappa = 1                      * kappa fixed
    * kappa = 4.43491                * SET THIS to fix kappa at MLE under M0
    * kappa = 4.39117                * SET THIS to fix kappa at MLE under M1
    * kappa = 5.08964                * SET THIS to fix kappa at MLE under M2
    * kappa = 4.89033                * SET THIS to fix kappa at MLE under M3
    * kappa = 4.22750                * SET THIS to fix kappa at MLE under M7
    * kappa = 4.87827                * SET THIS to fix kappa at MLE under M8

  fix_omega = 0                      * omega to be estimated
      omega = 5                      * initial omega

    * ncatG = 3                      * SET THIS for 3 site categories under M3
    * ncatG = 10                     * SET THIS for 10 of site categories under M7 and M8

fix_blength = 2                      * fixed branch lengths from tree file
```

A note about exercise 4 run-times (OUTDATED)

Model	Full run-time	Exercise run time
M0	01:09:42	00:01:02
M1a	01:50:50	00:02:01
M2a	02:49:49	00:10:18
M3	04:00:51	00:20:53
M7	07:45:39	00:17:37
M8	14:43:38	00:34:55

- Try running models M0 and M1a now
- Run the rest of the models overnight
- I am here for both weeks; see me if you have any questions

Complete this table (If you forget what to do, there is a "step-by-step" guide on the course web-site.)

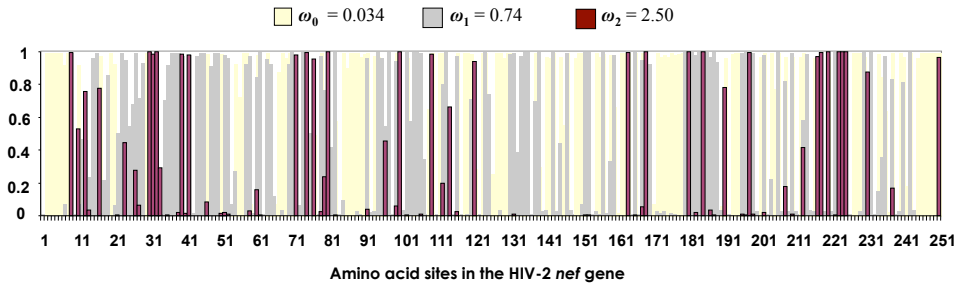
Table E4: Parameter estimates and likelihood scores under models of variable ω ratios among sites for HIV-2 *nef* genes.

Nested model pairs	d_N/d_S [b]	Parameter estimates [c]	PSS [d]	ℓ
M0: one-ratio (1) [a]	?	ω = ?	N.A.	?
M3: discrete (5)	?	p_0 = ?, p_1 = ?, (p_2 = ?) ω_0 = ?, ω_1 = ?, ω_2 = ?	? (?)	?
M1: neutral (1)	?	p_0 = ?, (p_1 = ?) ω_0 = ?, (ω_1 = 1)	N.A.	?
M2: selection (3)	?	p_0= ?, p_1 = ?, (p_2 = ?) ω_0 = ?, (ω_1 = 1), ω_2 = ?	? (?)	?
M7: beta (2)	?	p = ?, q = ?	N.A.	?
M8: beta&ω (4)	?	p_0 = ? (p_1 = ?) p = ?, q = ?, ω = ?	? (?)	?

[a] The number after the model code, in parentheses, is the number of free parameters in the ω distribution.
[b] This d_N/d_S ratio is an average over all sites in the HIV-2 *nef* gene alignment.
[c] Parameters in parentheses are not free parameters.
[d] PSS is the number of positive selection sites (NEB). The first number is the PSS with posterior probabilities > 50%. The second number (in parentheses) is the PSS with posterior probabilities > 95%.

NOTE: Codeml now implements models M1a and M2a !

Reproduce this plot (use the "rst" file generated by M3)



Rasmus Nielsen
Editor

Statistical Methods
in Molecular Evolution

Springer

5

Maximum Likelihood Methods for Detecting Adaptive Protein Evolution

Joseph P. Bielawski[1] and Ziheng Yang[2]

[1] Department of Biology, Dalhousie University, Halifax, Nova Scotia B3H 4J1, Canada, j.bielawski@dal.ca
[2] Department of Biology, University College London, Gower Street, London WC1E 6BT, United Kingdom, z.yang@ucl.ac.uk

5.1 Introduction

Proteins evolve; the genes encoding them undergo mutation, and the evolutionary fate of the new mutation is determined by random genetic drift as well as purifying or positive (Darwinian) selection. The ability to analyze this process was realized in the late 1970s when techniques to measure genetic variation at the sequence level were developed. The arrival of molecular sequence data also intensified the debate concerning the relative importance of neutral drift and positive selection to the process of molecular evolution [17]. Ever since, there has been considerable interest in documenting cases of molecular adaptation. Despite a spectacular increase in the amount of available nucleotide sequence data since the 1970s, the number of such well-established cases is still relatively small [9, 38]. This is largely due to the difficulty in developing powerful statistical tests for adaptive molecular evolution. Although several powerful tests for nonneutral evolution have been developed [33], significant results under such tests do not necessarily indicate evolution by positive selection.

A powerful approach to detecting molecular evolution by positive selection derives from comparison of the relative rates of synonymous and nonsynonymous substitutions [22]. Synonymous mutations do not change the amino acid sequence; hence their substitution rate (d_S) is neutral with respect to selective pressure on the protein product of a gene. Nonsynonymous mutations do change the amino acid sequence, so their substitution rate (d_N) is a function of selective pressure on the protein. The ratio of these rates ($\omega = d_N/d_S$) is a measure of selective pressure. For example, if nonsynonymous mutations are deleterious, purifying selection will reduce their fixation rate and d_N/d_S will be less than 1, whereas if nonsynonymous mutations are advantageous, they will be fixed at a higher rate than synonymous mutations, and d_N/d_S will be greater than 1. A d_N/d_S ratio equal to one is consistent with neutral evolution.