jModelTest 2 Manual v0.1.10

Diego Darriba, David Posada

Contents

1 Overview **2**
1.1 Download . . . 2
1.2 Citation . . . 2
1.3 Disclaimer . . . 2
1.4 Last Updates . . . 3

2 Getting Started **5**
2.1 Operating Systems . . . 5
2.2 Working with the repository . . . 5
2.3 User interfaces . . . 5
2.4 High Performance Environments . . . 11

3 Graphical User Interface **13**
3.1 Launching the Graphical User Interface . . . 13
3.2 Menu description . . . 14

4 Command Line Arguments **15**

5 Global Configuration **18**
5.1 Logging properties . . . 19
5.2 PhyML binary properties . . . 19
5.3 Hybrid shared-distributed memory execution properties . . . 20

6 Common Use Cases **21**
6.1 Converting Alignment Files . . . 21
6.2 Basic Model Selection . . . 21
6.3 Loading Checkpointing Files . . . 21

7 Theoretical Background **23**
7.1 Models of nucleotide substitution . . . 23
7.2 Sequential Likelihood Ratio Tests (sLRT) . . . 23
7.3 Hierarchical Likelihood Ratio Tests (hLRT) . . . 24
7.4 Dynamical Likelihood Ratio Tests (dLRT) . . . 24
7.5 Information Criteria . . . 25
7.6 Model Uncertainty . . . 26
7.7 Model Averaging . . . 26
7.8 Model Averaged Phylogeny . . . 26
7.9 Parameter Importance . . . 26

1 Overview

jModelTest is a tool to carry out statistical selection of best-fit models of nucleotide substitution. It implements five different model selection strategies: hierarchical and dynamical likelihood ratio tests (hLRT and dLRT), Akaike and Bayesian information criteria (AIC and BIC), and a decision theory method (DT). It also provides estimates of model selection uncertainty, parameter importances and model-averaged parameter estimates, including model-averaged tree topologies. jModelTest 2 includes High Performance Computing (HPC) capabilities and additional features like new strategies for tree optimization, model-averaged phylogenetic trees (both topology and branch lenght), heuristic filtering and automatic logging of user activity.

1.1 Download

The main project webpage is located at GitHub: https://github.com/ddarriba/jmodeltest2.

New distributions of jModelTest will be hosted in GitHub releases and google drive.

- https://github.com/ddarriba/jmodeltest2/releases
- https://drive.google.com/folderview?id=0ByrkKOPtF_n_OUs3d0dNcnJPYXM#list

Please use the jModelTest discussion group for any question:

- http://groups.google.com/group/jmodeltest.

1.2 Citation

When using jModelTest you should cite all these:

- Darriba D, Taboada GL, Doallo R, Posada D. 2012. jModelTest 2: more models, new heuristics and parallel computing. Nature Methods 9(8), 772.
- Guindon S and Gascuel O (2003). A simple, fast and accurate method to estimate large phylogenies by maximum-likelihood". Systematic Biology 52: 696-704.

1.3 Disclaimer

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 3 of the License, or (at your option) any later version. This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details. You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 59 Temple Place - Suite 330, Boston, MA 02111-1307, USA. The jModelTest distribution includes Phyml executables.

These programs are protected by their own license and conditions, and using jModelTest implies agreeing with those conditions as well.

1.4 Last Updates

- 3 Mar 2016 Version 2.1.10 Revision 20160303
  - Fixed bug with sequences where the 8-char name prefixes are equal
  - Added warning when the logging is disabled on runtime
  - Added win32 PhyML binary version to compatibility list
- 15 Jan 2016 - Version 2.1.9
  - Added automatic search for PhyML binary in /usr/bin
  - Removed non-ASCII characters
  - Disable logging if writing is not possible
  - Merge GUI images into jarfile
- 20 Oct 2015 - Version 2.1.8
  - Removed ReadSeq dependency
  - Fixed warnings
  - Updated prottest jarfile to v3.4
- 20 Feb 2015 - Version 2.1.7
  - Fixed bug in ML tree search operation. Console version was using NNI moves instead of "BEST" by default.
- 20 Nov 2014 - Version 2.1.7
  - Fixed bug with special characters in paths
  - Added initial check of PhyML binaries
  - Added notification in case AICc produces negative values
- 06 Aug 2014 - Version 2.1.6
  - Added confirmation window when cancelling running jobs in the GUI
  - Added automatic checkpointing files generation
  - Added "-ckp" argument for loading checkpointing files
- 05 Apr 2014 - Version 2.1.5
  - Updated OS X binary
  - Fixed bug with computation of JC model for "fixed" topology
  - Fixed bug with DT criterion computation
  - Added "-n" argument for naming executions (the name is included in the log filenames)
  - Added "-getphylip" argument for converting alignments into PHYLIP format with ALTER
  - Fixed bug in PhyML logging in GUI. Added a unique ID for every model in the log file
  - Added PAUP* block into log files if required ("-w" argument)
  - Added more verbose error messages
- 10 Jul 2013 - Version 2.1.4
  - Added phyml auto-logging.
  - Added phyml command lines for best-fit models.
  - Added phyml log tab in the GUI.
  - Removed sample size modes (and "-n" argument). Sample size is fixed to alignment size.
  - Fixed bug with relative paths when calling from a different path.
  - Fixed typos in the GUI.
- 05 Mar 2013 - Version 2.1.3
  - Fixed bug with PAUP'*' command block.
  - Added the possibility to change Inforation Criterion used with the clustering algorithm for the 203 matrices.
  - Changed "-o" argument for the hypothesis order into "-O"
  - Added "-o" argument for forwarding the standard output to a file: -o FILENAME
- 01 Jan 2013 Version 2.1.2 - Revision 20130103
  - Fixed bug in paths with whitespaces.
  - Updated PhyML binaries.
- 31 Jul 2012 Version 2.1.1 - Revision 20120731
  - Fixed bug with hLRT selection when attempting to use a user-defined topology.
- 11 Mar 2012 Version 2.1 - Revision 20120511
  - Major updates:

    * Exhaustive GTR submodels: All the 203 different partitions of the GTR rate matrix can be included in the candidate set of models. When combined with rate variation (+I,+G, +I+G) and equal/unequal base frequencies the total number of possible models is 203 x 8 = 1624.
    * Hill climbing hierarchical clustering: Calculating the likelihood score for a large number of models can be extremely time-consuming. This hill-climbing algorithm implements a hierarchical clustering to search for the best-fit models within the full set of 1624 models, but optimizing at most 288 models while maintaining model selection accuracy.
    * Heuristic filtering: Heuristic reduction of the candidate models set based on a similarity filtering threshold among the GTR rates and the estimates of among-site rate variation.
    * Absolute model fit: Information criterion distances can be calculated for the best-fit model against the unconstrained multinomial model (based on site pattern frequencies). This is computed by default when the alignment does not contain missing data/ambiguities, but can also be approximated otherwise.
    * Topological summary: Tree topologies supported by the different candidate models are summarized in the html log, including confidence intervals constructed from cumulative models weights, plus Robinson-Foulds and Euclidean distances to the best-fit tree for each.
  - Minor updates:
    * Corrected a bug in the fixed BIONJ-JC starting topology. F81+I+G was executed instead of JC.
    * "Best" is now the default tree search operation instead of NNI. "Best" computes both NNI and SPR algorithms and selects the best of them.
    * User can select the number of threads from GUI.

- 1 Feb 2012 - Version 2.0.2
  - Added a selection summary at the end of the console output.
  - Corrected the table header in the DT results frame (sorting).
  - Corrected a bug in DT Criterion where selection could not take place with large alignments.
  - Corrected a bug with command console version, where the execution crashed with certain arguments.
  - Unified LOCALE for English format.

- 2 Nov 2011 - Version 2.0.1
  - Improved thread scheduling algorithm.
  - OpenMP phyml patch for hybrid execution.
  - New argument (machinesfile) for hybrid execution on heterogeneous architectures, or heterogeneous resources distribution.

- 13 Oct 2011 - Revision 20111013
  - Added conf/jmodeltest.conf file, where you can: Enable/Disable the automatic logging:
    You might be running a huge dataset and you don't want to generate hundreds or thousands of log files.
    Set the PhyML binaries location:
    If you already have installed PhyML in your machine, you can setup jModelTest for use your own binaries.
  - Enhanced the html log output.

2 Getting Started

2.1 Operating Systems

Since jModelTest is a Java application, it can be used in every OS that can execute a Java Runtime Environment (JRE). The most common Operating Systems and many other include a JRE (OpenJDK, Sun JRE, ...), or at least it is possible to download one. However, jModelTest depends on third-party binaries (PhyML), that are distributed for Windows, Linux and OsX, and it is even possible to download PhyML sources (http://code.google.com/p/phy and compile them for a particular architecture.

2.2 Working with the repository

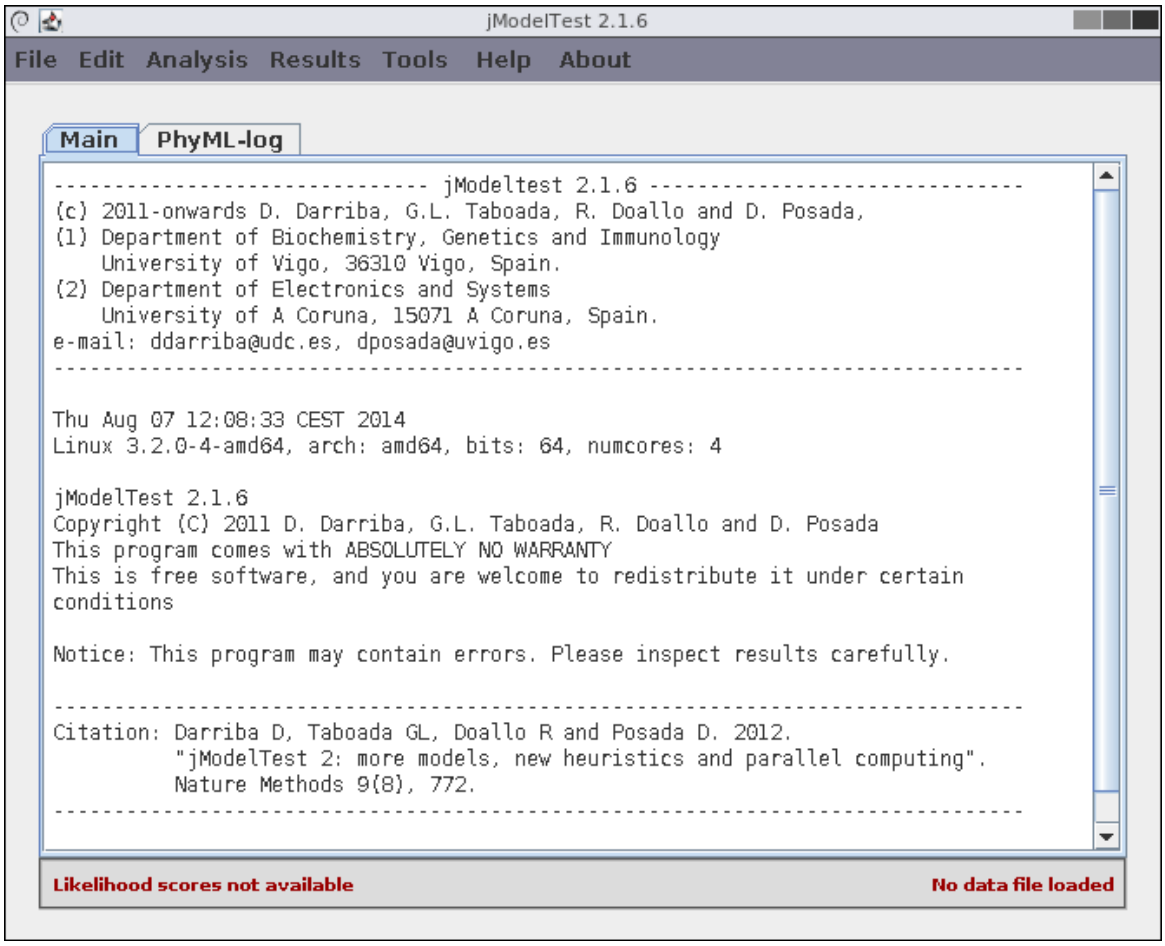
This tool is distributed under GPL v3 license. The source code is freely available at google code repository. You can checkout the repository at `http://code.google.com/p/jmodeltest2/source`.

2.3 User interfaces

jModelTest can be executed from two different user interfaces, GUI or Console. The Graphical User Interface (GUI) is intended for execution on common desktop computers with multicore processors -most users will probably use this. On the other hand, HPC environments, like multicore clusters, require a non-interactive processing (batch processes), so jModelTest has to be executed from the Command Console Interface. Results are given in plain text format, but an html log is also created.

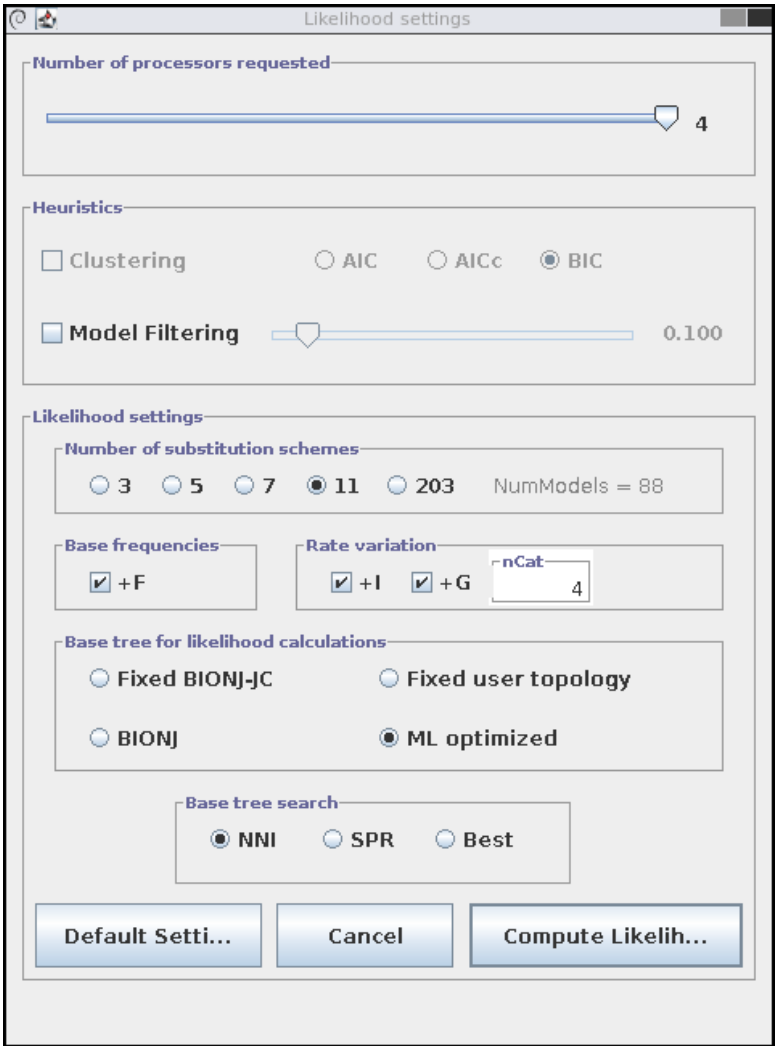
2.3.1 Graphical User Interface

1. Execute the script for the Graphical User Interface (runjmodeltest-gui.sh). The main jModelTest frame should pop up on the screen:

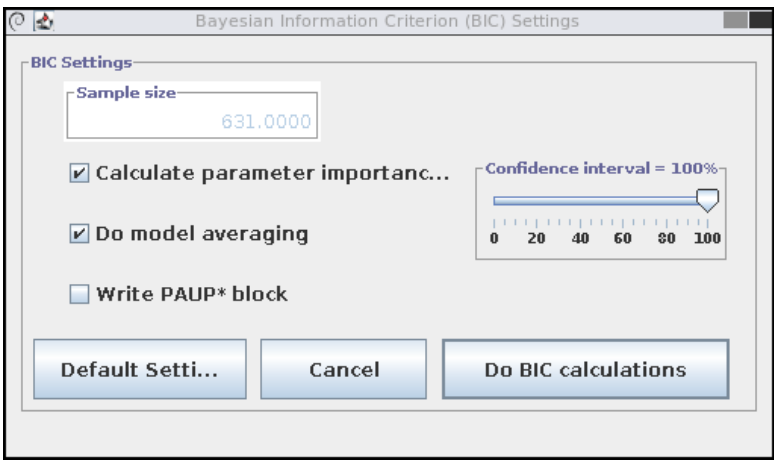


2. Load an input alignment file using the **File/Load Alignment** option.

3. Go to **Analysis/Compute Likelihood Scores** and select the candidate models and the options for model optimization (optionally you can set a base topology from a file). Press Enter or the **Compute Likelihoods** button.

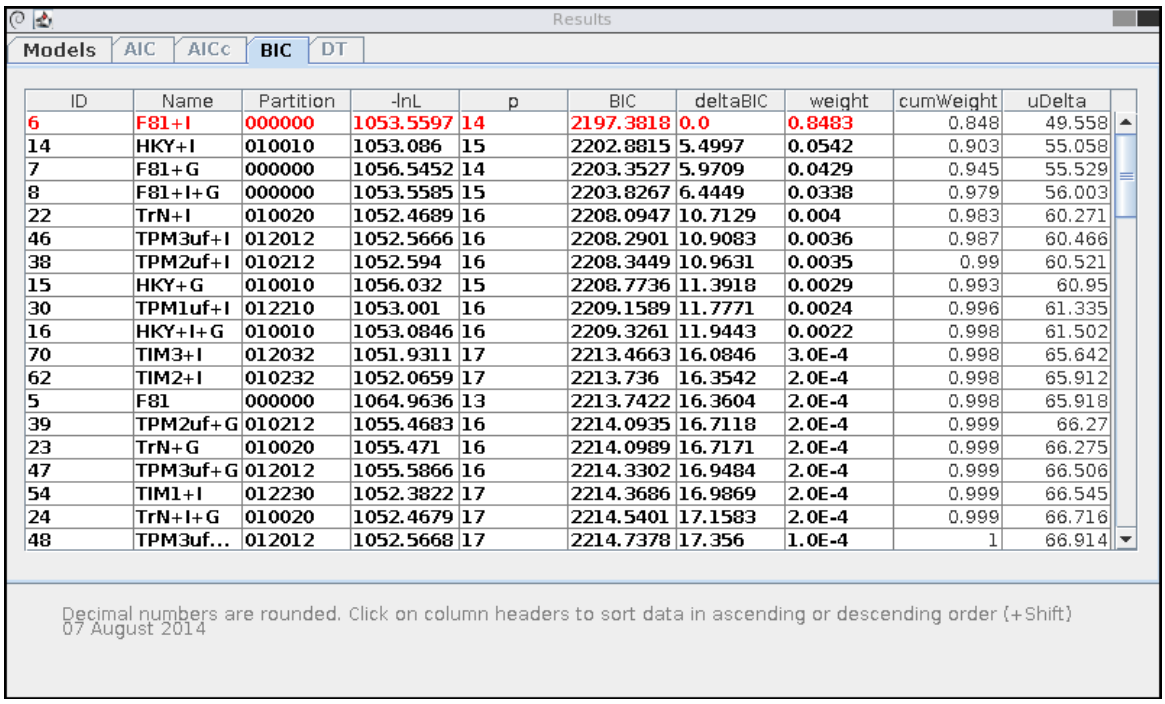


4. Perform statistical selection among the optimized models. For example, we can calculate the Bayesian Information Criterion using **Analysis/Do BIC calculations...** option, or any other. You can find a Criteria comparison in terms of accuracy in the supplementary material of the jModelTest publication.



The results will be shown in the main console.

5. Take a look at the results table in **Results/Show results table**. Best model is the one with the lowest criterion value (BIC column in the example) and therefore delta = 0.



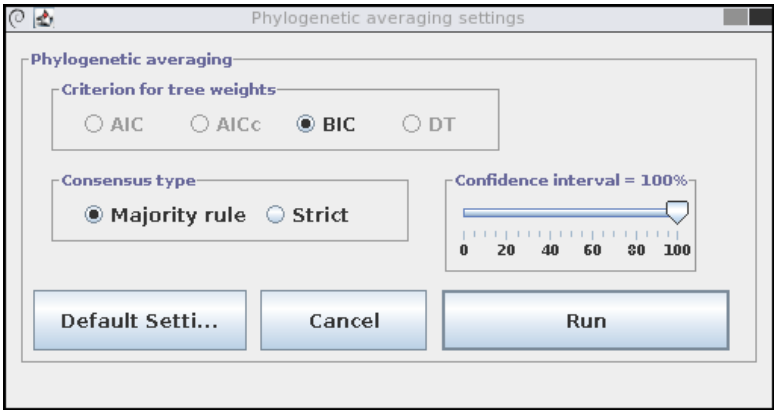
Results

Models | AIC | AICc | BIC | DT

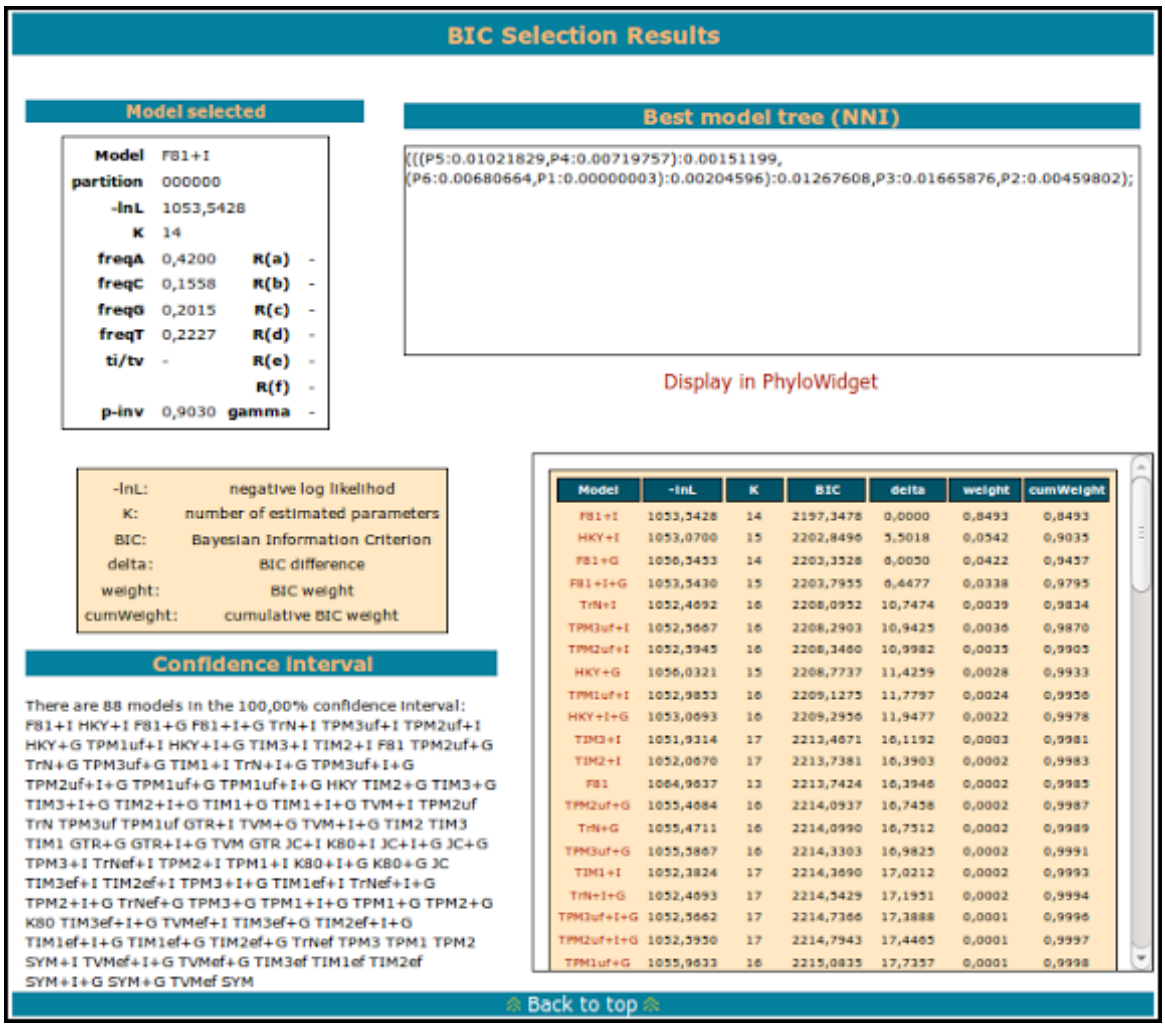
ID	Name	Partition	-lnL	p	BIC	deltaBIC	weight	cumWeight	uDelta
6	F81+I	000000	1053.5597	14	2197.3818	0.0	0.8483	0.848	49.558
14	HKY+I	010010	1053.086	15	2202.8815	5.4997	0.0542	0.903	55.058
7	F81+G	000000	1056.5452	14	2203.3527	5.9709	0.0429	0.945	55.529
8	F81+I+G	000000	1053.5585	15	2203.8267	6.4449	0.0338	0.979	56.003
22	TrN+I	010020	1052.4689	16	2208.0947	10.7129	0.004	0.983	60.271
46	TPM3uf+I	012012	1052.5666	16	2208.2901	10.9083	0.0036	0.987	60.466
38	TPM2uf+I	010212	1052.594	16	2208.3449	10.9631	0.0035	0.99	60.521
15	HKY+G	010010	1056.032	15	2208.7736	11.3918	0.0029	0.993	60.95
30	TPM1uf+I	012210	1053.001	16	2209.1589	11.7771	0.0024	0.996	61.335
16	HKY+I+G	010010	1053.0846	16	2209.3261	11.9443	0.0022	0.998	61.502
70	TIM3+I	012032	1051.9311	17	2213.4663	16.0846	3.0E-4	0.998	65.642
62	TIM2+I	010232	1052.0659	17	2213.736	16.3542	2.0E-4	0.998	65.912
5	F81	000000	1064.9636	13	2213.7422	16.3604	2.0E-4	0.998	65.918
39	TPM2uf+G	010212	1055.4683	16	2214.0935	16.7118	2.0E-4	0.999	66.27
23	TrN+G	010020	1055.471	16	2214.0989	16.7171	2.0E-4	0.999	66.275
47	TPM3uf+G	012012	1055.5866	16	2214.3302	16.9484	2.0E-4	0.999	66.506
54	TIM1+I	012230	1052.3822	17	2214.3686	16.9869	2.0E-4	0.999	66.545
24	TrN+I+G	010020	1052.4679	17	2214.5401	17.1583	2.0E-4	0.999	66.716
48	TPM3uf...	012012	1052.5668	17	2214.7378	17.356	1.0E-4	1	66.914

Decimal numbers are rounded. Click on column headers to sort data in ascending or descending order (+Shift)
07 August 2014

6. Build a consensus tree from a given selection criteria using **Analysis/Model-averaged phylogeny**:



7. Finally, you can save the results displayed in the main console using **Edit/Save console**. Alternatively, you can get a formatted HTML document using **Results/Build HTML log**:



BIC Selection Results

Model selected

Model	F81+I		
partition	000000		
-lnL	1053,5428		
K	14		
freqA	0,4200	R(a)	-
freqC	0,1558	R(b)	-
freqG	0,2015	R(c)	-
freqT	0,2227	R(d)	-
ti/tv	-	R(e)	-
		R(f)	-
p-inv	0,9030	gamma	-

Best model tree (NNI)

(((P5:0.01021829,P4:0.00719757):0.00151199,
(P6:0.00680664,P1:0.00000003):0.00204596):0.01267608,P3:0.01665876,P2:0.00459802);

Display in PhyloWidget

-lnL:	negative log likelihod
K:	number of estimated parameters
BIC:	Bayesian Information Criterion
delta:	BIC difference
weight:	BIC weight
cumWeight:	cumulative BIC weight

Model	-lnL	K	BIC	delta	weight	cumWeight
F81+I	1053,5428	14	2197,3478	0,0000	0,8493	0,8493
HKY+I	1053,0700	15	2202,8496	5,5018	0,0542	0,9035
F81+G	1056,5453	14	2203,3528	6,0050	0,0422	0,9457
F81+I+G	1053,5430	15	2203,7955	6,4477	0,0338	0,9795
TrN+I	1052,4692	16	2208,0952	10,7474	0,0039	0,9834
TPM3uf+I	1052,5667	16	2208,2903	10,9425	0,0036	0,9870
TPM2uf+I	1052,5945	16	2208,3460	10,9982	0,0035	0,9905
HKY+G	1056,0321	15	2208,7737	11,4259	0,0028	0,9933
TPM1uf+I	1052,9853	16	2209,1275	11,7797	0,0024	0,9956
HKY+I+G	1053,0693	16	2209,2956	11,9477	0,0022	0,9978
TIM3+I	1051,9314	17	2213,4671	16,1192	0,0003	0,9981
TIM2+I	1052,0670	17	2213,7381	16,3903	0,0002	0,9983
F81	1064,9637	13	2213,7424	16,3946	0,0002	0,9985
TPM2uf+G	1055,4684	16	2214,0937	16,7458	0,0002	0,9987
TrN+G	1055,4711	16	2214,0990	16,7512	0,0002	0,9989
TPM3uf+G	1055,5867	16	2214,3303	16,9825	0,0002	0,9991
TIM1+I	1052,3824	17	2214,3690	17,0212	0,0002	0,9993
TrN+I+G	1052,4693	17	2214,5429	17,1951	0,0002	0,9994
TPM3uf+I+G	1052,5662	17	2214,7366	17,3888	0,0001	0,9996
TPM2uf+I+G	1052,5950	17	2214,7943	17,4465	0,0001	0,9997
TPM1uf+G	1055,9633	16	2215,0835	17,7357	0,0001	0,9998

Confidence Interval

There are 88 models in the 100,00% confidence interval:
F81+I HKY+I F81+G F81+I+G TrN+I TPM3uf+I TPM2uf+I HKY+G TPM1uf+I HKY+I+G TIM3+I TIM2+I F81 TPM2uf+G TrN+G TPM3uf+G TIM1+I TrN+I+G TPM3uf+I+G TPM2uf+I+G TPM1uf+G TPM1uf+I+G HKY TIM2+G TIM3+G TIM3+I+G TIM2+I+G TIM1+G TIM1+I+G TVM+I TPM2uf TrN TPM3uf TPM1uf GTR+I TVM+G TVM+I+G TIM2 TIM3 TIM1 GTR+G GTR+I+G TVM GTR JC+I K80+I JC+I+G JC+G TPM3+I TrNef+I TPM2+I TPM1+I K80+I+G K80+G JC TIM3ef+I TIM2ef+I TPM3+I+G TIM1ef+I TrNef+I+G TPM2+I+G TrNef+G TPM3+G TPM1+I+G TPM1+G TPM2+G K80 TIM3ef+I+G TVMef+I TIM3ef+G TIM2ef+I+G TIM1ef+I+G TIM1ef+G TIM2ef+G TrNef TPM3 TPM1 TPM2 SYM+I TVMef+I+G TVMef+G TIM3ef TIM1ef TIM2ef SYM+I+G SYM+G TVMef SYM

Back to top

Take a look at Section 3 for further information.

2.3.2 Command Console Interface

1. Execute the following command line:

```
$ java -jar jModelTest.jar -d example-data/aP6.fas -g 4 -i -f -AIC -BIC -a
```

This will test all 88 models (gamma models with 4 rate categories), and then perform the model selection using Akaike (AIC) and Bayesian (BIC) criteria, calculating also a model averaged phylogeny (-a).

See Section 4 for information about supported arguments.

2. This will generate the following output:

   (a) Header:

```
------------------------------ jModeltest 2.0 ------------------------------
(c) 2011-onwards Diego Darriba, David Posada,
Department of Biochemistry, Genetics and Immunology
University of Vigo, 36310 Vigo, Spain. e-mail: ddarriba@udc.es, dposada@uvigo.es
----------------------------------------------------------------------------
Wed Oct 05 12:56:47 CEST 2011
Linux 2.6.38-11-generic-pae, arch: i386, bits: 32, numcores: 2

jModelTest 2.0  Copyright (C) 2011 Diego Darriba, David Posada
This program comes with ABSOLUTELY NO WARRANTY
This is free software, and you are welcome to redistribute it
under certain conditions

Notice: This program may contain errors. Please inspect results carefully.
```

(b) Execution options:

```
Arguments = -d example-data/aP6.fas -g 4 -i -f -AIC -BIC -a

Reading data file "aP6.fas"... OK.
  number of sequences: 6
  number of sites: 631


-----------------------------------------------------------------
*                                                               *
*          COMPUTATION OF LIKELIHOOD SCORES WITH PHYML          *
*                                                               *
-----------------------------------------------------------------

::Settings::
 Phyml version = 3.0
 Phyml binary = PhyML_3.0_linux32
 Candidate models = 24
  number of substitution schemes = 3
  including models with equal/unequal base frequencies (+F)
  including models with/without a proportion of invariable sites (+I)
  including models with/without rate variation among sites (+G) (nCat = 4)
 Optimized free parameters (K) = substitution parameters + 9 branch lengths + topology
 Base tree for likelihood calculations = ML tree
 Tree topology search operation = NNI
computing likelihood scores for 24 models with Phyml 3.0
```

(c) Real time optimization results (progress):

```
::Progress::

Model        Exec. Time     Total Time     -lnL
-------------------------------------------------------------------------
JC     00h:00:00:01   00h:00:00:01   1114,9772
JC+G     00h:00:00:04   00h:00:00:05   1106,4431
...
GTR+G    00h:00:00:06   00h:00:06:07   1054,7203
GTR+I+G    00h:00:01:02   00h:00:07:05   1051,8403
```

(d) Sorted and complete optimization results:

```
   Model = JC
   partition = 000000
   -lnL = 1114.9772
   K = 10

   Model = JC+I
   partition = 000000
   -lnL = 1103.1113
   K = 11
   p-inv = 0.9080

...

   Model = GTR+I+G
   partition = 012345
   -lnL = 1051.8403
   K = 20
   freqA = 0.4235
   freqC = 0.1520
   freqG = 0.2022
   freqT = 0.2224
   R(a) [AC] =  0.8709
   R(b) [AG] =  0.4152
   R(c) [AT] =  0.6049
   R(d) [CG] =  1.2523
   R(e) [CT] =  0.9482
   R(f) [GT] =  1.0000
   p-inv = 0.5940
   gamma shape = 0.0120
```

```
Computation of likelihood scores completed. It took 00h:00:07:05.
```

(e) Selected Information Criteria (best model and all models sorted according to each criterion):

```
-----------------------------------------------------------------
*                                                               *
*                AKAIKE INFORMATION CRITERION (AIC)             *
*                                                               *
-----------------------------------------------------------------

 Model selected:
   Model = F81+I
   partition = 000000
   -lnL = 1053.5428
   K = 14
   freqA = 0.4200
   freqC = 0.1558
   freqG = 0.2015
   freqT = 0.2227
   p-inv = 0.9030

ML tree (NNI) for the best AIC model = (((P5:0.01021829,P4:0.00719757):0.00151199,(P6
    :0.00680664,P1:0.00000003):0.00204596):0.01267608,P3:0.01665876,P2:0.00459802);


* AIC MODEL SELECTION : Selection uncertainty

Model              -lnL    K         AIC       delta       weight cumWeight
-----------------------------------------------------------------------------
F81+I          1053.5428   14   2135.0855      0.0000       0.4332     0.4332
HKY+I          1053.0700   15   2136.1401      1.0545       0.2557     0.6890
F81+I+G        1053.5430   15   2137.0859      2.0004       0.1594     0.8483
...
K80            1114.5049   11   2251.0098    115.9243    2.91e-026     1.0000
SYM            1114.4117   15   2258.8235    123.7380    5.85e-028     1.0000
-----------------------------------------------------------------------------
-lnL: negative log likelihod
 K: number of estimated parameters
 AIC: Akaike Information Criterion
 delta: AIC difference
 weight:  AIC weight
 cumWeight: cumulative AIC weight


* AIC MODEL SELECTION : Confidence interval

There are 24 models in the 100% confidence interval: [ F81+I HKY+I F81+I+G HKY+I+G F81+G
    GTR+I HKY+G GTR+I+G GTR+G F81 HKY GTR JC+I K80+I JC+I+G K80+I+G JC+G K80+G SYM+I SYM+I
    +G SYM+G JC K80 SYM ]
```

(f) Consensus tree of the optimized phylogenies using the criterion weights:

```
-----------------------------------------------------------------
*                                                               *
*                     MODEL AVERAGED PHYLOGENY                  *
*                                                               *
-----------------------------------------------------------------

Selection criterion: . . . . AIC
Confidence interval: . . . . 1.00
Consensus type:. . . . . . . 50% majority rule


Using 24 models in the 1.00 confidence interval = F81+I HKY+I F81+I+G HKY+I+G F81+G GTR+I
    HKY+G GTR+I+G GTR+G F81 HKY GTR JC+I K80+I JC+I+G K80+I+G JC+G K80+G SYM+I SYM+I+G SYM
    +G JC K80 SYM

Bipartitions included in the consensus tree

    123456
    ****** ( 1.0 )
    ****-- ( 1.0 )
    **---- ( 0.94244 )
```

```
    --**-- ( 1.0 )


                          +-----------6 P4
                       +-8
                       | +----------------5 P5
+---------------------9
|                      |  +-4 P1
|                      +--7
|                         +----------3 P6
|
+------2 P2
|
+---------------------------1 P3


(P3:0.016613,P2:0.004598,((P6:0.006790,P1:0.000000)1.00:0.002046,(P5:0.010191,P4:0.007198)
    0.94:0.001510)1.00:0.012665);

Note: this tree is unrooted. Branch lengths are the expected number of substitutions per
    site. Labels next to parentheses represent phylogenetic uncertainty due to model
    selection (see documentation)
```

(g) Also a HTML log is automatically stored in the "log" directory.

2.4 High Performance Environments

2.4.1 Shared memory architectures (multicore systems)

Both the GUI and Console interfaces can be used for shared memory architectures. See Graphical User Interface or Command Console Interface. In some dedicated HPC environments you can only use the console interface, for example when using a bath-queuing system like Oracle Grid Engine. Additionally, in the console version you can specify the number of threads you want to use using the -tr" option. By default, the total number of cores in the machine is used.

2.4.2 Distributed memory architectures (HPC clusters)

1. Besides the multithreading support, it is possible to run jModelTest in a cluster. This feature has been implemented using a Java message-passing (MPJ) library, MPJ Express (http://mpj-express.org/). To execute jModelTest in a cluster environment you have to:

```
$ export $JMODELTEST_HOME=[path_to_jModelTest]
$ cd $JMODELTEST_HOME
$ tar zvxf mpj.tar.gz
$ export MPJ_HOME=$JMODELTEST_HOME/mpj
$ export PATH=$MPJ_HOME/bin:$PATH
$ cp $JMODELTEST_HOME/extra/machines $JMODELTEST_HOME
```

   You can also add the last two lines to /.bashrc to automatically set these variables at console startup.

2. $JMODELTEST_HOME/machines file contains the set of computing nodes where the mpj processes will be executed. By default it points to the localhost machine, so you should change it if you want to run a parallel execution over a cluster machine, just writing on each line the particular computing nodes (e.g. see filecluster8.conf.template).

3. Start the MPJ Express daemons:

```
$ mpjboot machines
```

   The application "mpjboot" should be in the execution path (it is located at $MPJ_HOME/bin). A ssh service must be running in the machines listed in the machines file. Moreover, port 10000 should be free. For more details refer to the MPJ Express documentation.

4. Run jModelTest. For this, the jModelTest distribution provides a bash script: 'runjmodeltest-cluster.sh'

   The basic syntax is:

   ./runjmodeltest-cluster.sh $NUMBER_OF_PROCESSORS $APPLICATION_PARAMETERS

```
$ ./runjmodeltest-cluster.sh 2 -d example-data/aP6.fas -s 11 -i -g 4 -f -AIC -a
```

3 Graphical User Interface

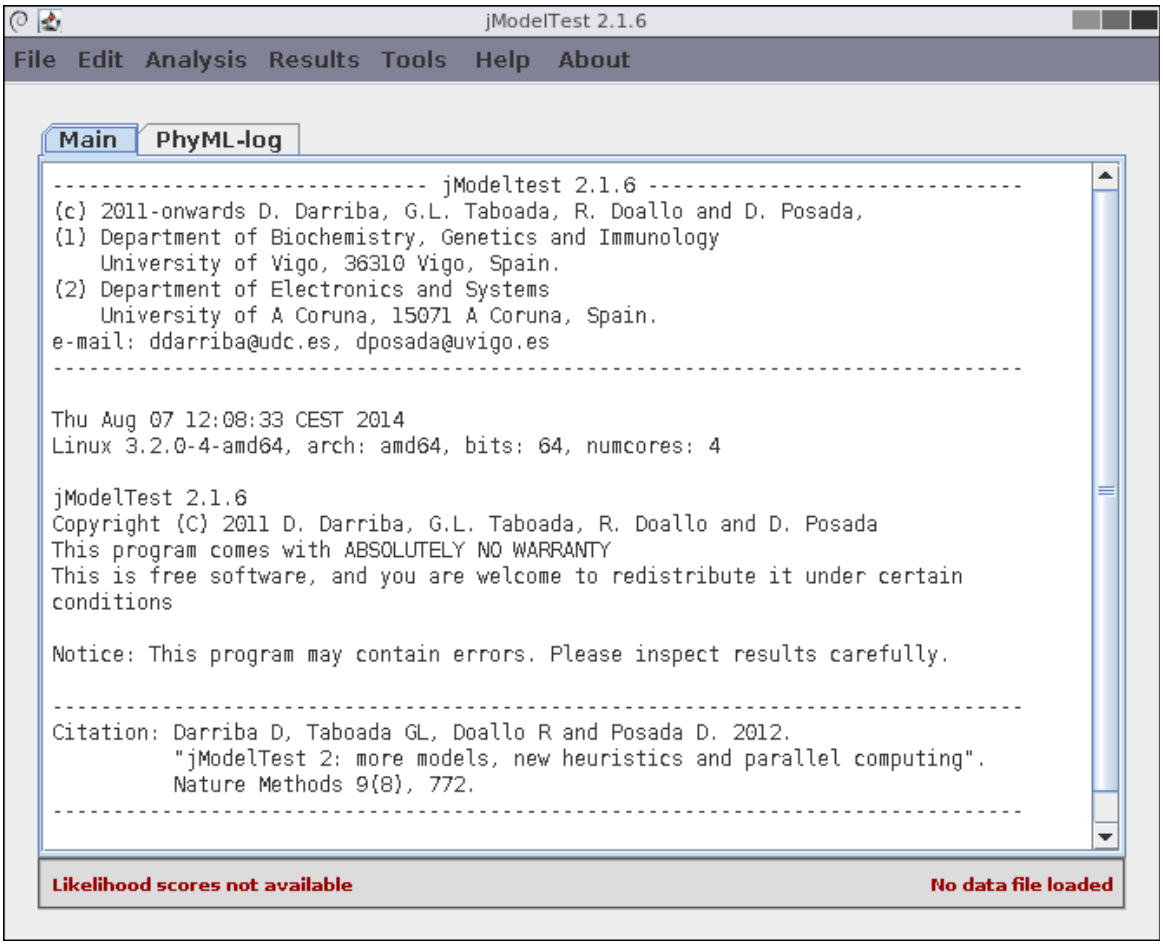
3.1 Launching the Graphical User Interface

The main distribution includes a script for launching the interface, *runjmodeltest-gui.sh*, located under the jModelTest home folder. Other possibility is running the following command line:

```
$ java -jar jModelTest.jar
```

Moreover, in Windows and MacOS X, it is often possible to double-click the jModelTest.jar file to launch the graphical interface.

The following window will show on the screen:



3.2 Menu description

Menu	Submenu	Description	Enabled
File			
	Load alignment	Load an input alignment	
	Load checkpoint file	Load a previous snapshot [a]	(i)
	Quit	Exit the program	
Analysis			
	Compute likelihood scores	Optimize the set of candidate models	(i)
	Do AIC calculations	Calculate Akaike Information Criterion	(ii)
	Do BIC calculations	Calculate Bayesian Information Criterion	(ii)
	Do DT calculations	Calculate Decision Theory	(ii)
	Do hLRT calculations	Calculate hierarchical likelihood ratio test	(ii) [b]
	Model-averaged phylogeny	Calculate the consensus tree	(iii & iv)
Results			
	Show results table	Show a table with the selection results	(ii)
	Build HTML log	Create an html webpage with the results	(ii)
Tools			
	LRT calculator	Likelihood Ratio Test for nexted models	

(i) After loading an alignment (ii) After computing the likelihood scores (iii) If the base tree is not fixed (iv) After calculating an Information Criterion

[a] See Section 6.3

[b] This test is only available for 3,5,7 and 11 substitution schemes and for fixed topologies (fixed BIONJ-JC tree or user-defined topology)

4 Command Line Arguments

- **-a**

  Estimate model-averaged phylogeny for each active criterion. See Section 7.8 for more details.

- **-AIC**

  Calculate the Akaike Information Criterion. See Section 7.5.1.

- **-AICc**

  Calculate the corrected Akaike Information Criterion. See Section 7.5.1.

- **-BIC**

  Calculate the Bayesian Information Criterion. See Section 7.5.2.

- **-DT**

  Calculate the decision theory criterion. See Section 7.5.3.

- **-c** confidenceInterval

  Sets the confidence interval for the model selection process (default is 100).

- **-d** inputFile

  Sets the input data file. jModelTest makes use of the ALTER library for converting several alignment formats to PHYLIP.

- **-dLRT**

  Perform dynamical likelihood ratio tests. See Section 7.4 for more details.

- **-f**

  Include models with unequals base frecuencies.

- **-g** numberOfRateCategories

  Include models with rate variation among sites and sets the number of categories. Usually 4 categories are enough.

- **-getPhylip**

  Converts the input file into phylip format and exits. For example, the following command will generate a new PHYLIP file named "input.nex.phy".

```
$java -jar jModelTest.jar -d input.nex -getPhylip
```

- **-G** threshold

  Heuristic search. Requires a threshold ¿ 0 (e.g., -G 0.1)

- **-h** confidenceInterval

  Sets the confidence level for the hLRTs (default is 0.01)

- **-help**

  Displays a help message

- **-hLRT**

  Perform hierarchical likelihood ratio tests. See Section 7.3 for more details.

- **-H**

  Information criterion for clustering search (AIC, AICc, BIC). (e.g., -H AIC) (default is BIC)

- **-i**

  Include models with a proportion invariable sites.

- **-machinesfile** machinesFile

  Gets the processors per host from a machines file (for MPI execution).

- **-n** logSuffix

  Execution name appended to the log filenames. By default, current time is used: yyyyMMddhhmmss.

- **-o** outputFile

  Redirects the output to a file.

- **-O** ftvwxgp

  Sets the hypothesis order for the hLRTs (e.g., -hLRT -O gpftv) (default is ftvwxgp)

  - **f** frequencies
  - **t** transition/transversion ratio
  - **v** 2ti4tv for subst=3 / 2ti for subst¿3
  - **w** 2tv
  - **x** 4tv
  - **g** gamma
  - **p** proportion of invariable sites

  See Section 7.3 for more details.

- **-p**

  Calculate the parameter importances. See Section 7.9.

- **-r**

  Backward selection for the hLRT (default is forward).

- **-s** 3—5—7—11—203

  Sets the number of substitution schemes.

  - **3** JC/F81, K80/HKY, SYM/GTR (used by default).
  - **5** JC/F81, K80/HKY, TrNef/TrN, TPM1/TPM1uf, SYM/GTR.
  - **7** JC/F81, K80/HKY, TrNef/TrN, TPM1/TPM1uf, TIM1ef/TIM1, TVMef/TVM, SYM/GTR.
  - **11** All models defined in Table 1.
  - **203** All possible GTR submatrices.

- **-S** NNI—SPR—BEST

  Defines the tree topology search operation option for Maximum-Likelihood search:

  - **NNI** Nearest Neighbour Interchange (fast).
  - **SPR** Subtree Pruning and Regrafting (slower).
  - **BEST** Best of NNI and SPR (slowest option) (used by default).

- **–set-local-config** configFile

  Allows the user to set a local configuration file in replacement of conf/jmodeltest.conf. See Section 5 for more details.

- **–set-property** propertyName=propertyValue

  Allows the user to set a especific value for a property in replacement of the existing parameter in conf/jmodeltest.conf. See Section 5 for more details.

  e.g., –set-property log-dir=myHome/myLogDirectory

- **-t** fixed—BIONJ—ML

  Base tree for likelihood calculations (e.g., -t BIONJ):

  - **fixed** Fixed BIONJ topology from JC model
  - **BIONJ** Neighbor-Joining topology for each model
  - **ML** Maximum Likelihood topology for each model (default)

- **-tr** numberOfThreads

  Number of threads to execute (default is the number of logical processors in the machine).

- **-u** treeFile

  Fixed tree for likelihood calculations defined by the user. If a user tree is defined with this command, -t argument is ignored.

- **-uLnL**

  Calculate delta AIC,AICc,BIC against unconstrained likelihood.

- **-v**

  Do model averaging and parameter importances. See Section 7.7.

- **-w**

  Prints out the PAUP block.

- **-z**

  Strict consensus type for model-averaged phylogeny (default is majority rule). See Section 7.8.

5 Global Configuration

jModelTest contains some global configuration parameters in the file conf/jmodeltest.conf. In case you are sharing the jModelTest distribution between multiple users, it is possible to set a local configuration file for your own using the **--set-local-config** argument in the command file. You can also change one or several properties by using the **--set-property** argument. See Page 16 for the reference about this commands.

For example:

```
$ java -jar jModelTest -d example-data/aP6.phy -f -i -g 4 -s 11 --set-property log-dir=myLogDir --
    set-property exe-dir=path/to/my/phyml
```

This file has the following content:

```
#######################################
# jModelTest Configuration properties #
#######################################

##########################################################
#                                                        #
# Automatic Logging                                      #
#                                                        #
# If html-logging is "enabled", every time the user runs #
# jModelTest, a new html log file will be created in the #
# log directory.                                         #
# If phyml-logging is "enabled", PhyML streams are saved #
# Default log directory is $JMODELTEST_HOME/log, but can #
# be modified using the log-dir property.                #
#                                                        #
##########################################################
checkpointing  = enabled
html-logging   = enabled
phyml-logging  = enabled
log-dir        = log

##########################################################
#                                                        #
# Phyml Binaries path                                    #
#                                                        #
# By default, jModelTest will search for the PhyML       #
# executables in $JMODELTEST_HOME/exe/phyml. User can    #
# define a different path, wether absolute (starting     #
# with '/' or 'C:\') or relative to $JMODELTEST_HOME     #
# directory using exe-dir property.                      #
#                                                        #
# If an usable version of PhyML is installed system-wide #
# (for example, from the Ubuntu/Debian repositories),    #
# the user can set 'global-phyml-exe' property to true   #
# and jModelTest will use the global binary instead of   #
# local ones.                                            #
#                                                        #
##########################################################
global-phyml-exe    = false
exe-dir             = exe/phyml

##########################################################
#                                                        #
# Thread Scheduling Configuration                        #
#                                                        #
# Properties below are specific properties for the       #
# thread scheduling behavior. Those are the default      #
# number of threads for executing each sort of model.    #
```

```
#                                                        #
# If the specified number of threads is higher than the  #
# total number of cores in the machine, the whole        #
# machine will be used for that models.                  #
#                                                        #
##########################################################
gamma-threads    = 4
inv-threads      = 2
uniform-threads  = 1
```

5.1 Logging properties

All the logs generated by jModelTest will be stored in the folder defined by the **log-dir** property. The tool generates 3 different kinds of log files:

- **PhyML logs**. The output of PhyML for every model optimization. This files are useful when an error occurs during the model optimization.
- **HTML logs**. The results of the model selection in html format. This provides an easy visualization of the results.
- **checkpoint files**. Snapshots are stored during the execution, and these can be used for restoring a previous run at the last stable point.

Using the properties one can enable or disable each log independently. If there is no **log-dir**, all logs will be disabled independently on their value. Thus, for example a system administrator could comment that property in the configuration file, and users can set their own log directory in the command line:

```
checkpointing  = enabled
html-logging   = enabled
phyml-logging  = enabled
# log-dir        = log     <-- This line is commented
```

Now setting the custom log directory will generate all the log files there, as long as those are enabled in the configuration file. Otherwise, no log will be generated.

Property	Values	Default
checkpointing	enabled/disabled	enabled
html-logging	enabled/disabled	enabled
phyml-logging	enabled/disabled	enabled
log-dir	logging directory	log/

5.2 PhyML binary properties

jModelTest distribution includes PhyML binaries in exe/phyml directory. However, it is possible that you already have PhyML installed in your system and for some reason you prefer to use that distribution. In such a case, setting the **global-phyml-exe** property to true, jModelTest will try to run a "phyml" command that is expected to be in the PATH. Also if you want to set a different path than default for finding the PhyML binaries you can change the **exe-dir** property.

In that directory, an executable file called "phyml" will have priority. If it does not exist, jModelTest will try to execute the binary called "PhyML_3.0_PLATFORM". For example, "PhyML_3.0_linux64". Thus, if you have a working phyml binary you can place it in the binaries directory with the name "phyml".

Property	Values	Default
global-phyml-exe	true/false	false
exe-dir	path to local PhyML	exe/phyml

TIP: A workaround for setting the global PhyML executable in case that its name differs from "phyml" is creating a symbolic link in the binaries directory. For example:

```
$ cd $JMODELTEST_HOME/exe/phyml
$ ln -s phyml `which $MY_PHYML_GLOBAL_EXECUTABLE`
```

5.3 Hybrid shared-distributed memory execution properties

The following properties are used only with the hybrid memory parallel execution. The dynamic shared memory scheduler can use a different number of threads for model optimization depending on the model parameters. For example, models with rate heterogeneity will take a longer time to optimize the parameters. This way, assigning a different number of threads used for the parallel optimization of each model will improve the efficiency by minimizing the parallel overhead.

Note that for enabling the hybrid memory parallelization you need to create your own PhyML binaries applying a patch directly to the PhyML source code and compiling it for you system. You can ask us about it.

The default values work fine for a hybrid execution using 8 and 16 core-nodes.

Property	**Values**	**Default**
gamma-threads	#threads for +G and +I+G models	4
inv-threads	#threads for +I models	2
uniform-threads	#threads for models without rate heterogeneity	1

6 Common Use Cases

6.1 Converting Alignment Files

jModelTest accepts several input alignment file formats. However, it makes use of the ALTER library for converting them into PHYLIP format, accepted by PhyML. If you want to validate your alignment, you can convert it into PHYLIP format using the "-getPhylip" argument. It will generate a new file appending ".phy" to the input alignment filename, and exit afterwards.

```
$ java -jar jModelTest -d example-data/aP6.fas -getPhylip
```

In case there is something wrong in the input file, it will exit with the description of the error.

6.2 Basic Model Selection

Although jModelTest have many options, most of the users would like to perform a model selection among the 11 substitution schemes, including models with unequal frequencies, gamma rate variation and a proportion of invariable sites. The following command produces this operation, shows the selection results under the 4 available criteria, computes the model-averaged phylogenies ("-a"), computes the parameters importance ("-v" and "-p") and writes the PAUP* block for the best-fit models ("-w"):

```
$ java -jar jModelTest -d example-data/aP6.fas -s 11 -f -i -g 4 -AIC -BIC -AICc -DT -p -a -w
```

Note that, by default, jModelTest uses Maximum-Likelihood topologies as the base trees for the model optimization, and checks both NNI and SPR algorithms for the topology search. This obtains the most accurate results, but it is also the most time consuming operation. According to the size of the input alignment, one can directly select one of the algorithms saving time in the computations. As a general rule, for a small number of taxa NNI algorithm would work better, as well as SPR is more suitable for a large number of taxa. The tree search operation can be set with "-S" argument (e.g., -t ML -S NNI).

6.3 Loading Checkpointing Files

By default, jModelTest saves ".ckp" checkpointing files in the log directory. In case of an error occurs, the user can start again the process minimizing the loss of computation. The user is in charge of selecting the checkpointing file and running again jModelTest with the same parameters of the previous execution. Otherwise the results might be wrong.

For finding the correct checkpointing file, if the execution had a user-defined name "-n argument", the checkpoing file will have the following format:

```
log/[sequenceFileName].[executionName].ckp
```

For example, the following command:

```
$ java -jar jModelTest -d example-data/aP6.fas -n myTest -s 11 -f -i -g 4 -BIC -AIC
```

Will generate the checkpointing file in $JMODELTEST_HOME/log/aP6.fas.myTest.ckp, and in case of a sudden error in the execution, it can be continued using:

```
$ java -jar jModelTest -d example-data/aP6.fas -n myTest -s 11 -f -i -g 4 -BIC -AIC -ckp log/aP6.fas
    .myTest.ckp
```

If no execution name was provided, it is automatically generated according to the current date and time with the following format: yyyyMMddhhmmss (e.g., if current time is 17:05:00 August 3 2014, the execution name is 20140803170500, and the checkpointing generated file is:

```
log/[sequenceFileName].20140803170500.ckp).
```

When using the GUI instead of the command console interface, the checkpointing file can be loaded using the menu item "File/Load checkpoint file", that becomes enabled right after loading the alignment.

From the GUI, one can choose between the different number of the substituion schemes in the execution settings window.

7 Theoretical Background

All phylogenetic methods make assumptions, whether explicit or implicit, about the process of DNA substitution [Felsenstein, 1988]. Consequently, all the methods of phylogenetic inference depend on their underlying substitution models. To have confidence in inferences it is necessary to have confidence in the models [Goldman, 1993b]. Because of this, it makes sense to justify the use of a particular model. Statistical model selection is one way of doing this. For a review of model selection in phylogenetics see Sullivan and Joyce [2005] and Johnson and Omland [2003]. The strategies includes in jModelTest include sequential likelihood ratio tests (LRTs), Akaike Information Criterion (AIC), Bayesian Information Criterion (BIC) and performance-based decision theory (DT).

7.1 Models of nucleotide substitution

Models of evolution are sets of assumptions about the process of nucleotide substitution. They describe the different probabilities of change from one nucleotide to another along a phylogenetic tree, allowing us to choose among different phylogenetic hypotheses to explain the data at hand. Comprehensive reviews of model of evolution are offered elsewhere. jmodeltest implementes all 203 types of reversible substitution matrices, with when combined with unequal/equal base frequencies, gamma-distributed among-site rate variation and a proportion of invariable sites makes a total of 1624 models. Some of the models have received names (see Table 1):

Table 1: Named substitution models jModelTest2 (a few of the 1624 possible). Any of these models can include invariable sites (+I), rate variation among sites (+G), or both (+I+G).

Model	Reference	Free param.	Base freq.	Substitution rates	Substitution code
JC	[Jukes and Cantor, 1969]	0	equal	AC=AG=AT=CG=CT=GT	000000
F81	[Felsenstein, 1981]	3	unequal	AC=AG=AT=CG=CT=GT	000000
K80	[Kimura, 1980]	1	equal	AC=AT=CG=GT;AG=GT	010010
HKY	[Hasegawa et al., 1985]	4	unequal	AC=AT=CG=GT;AG=GT	010010
TrNef	[Tamura and Nei, 1993]	2	equal	AC=AT=CG=GT;AG;GT	010020
TrN	[Tamura and Nei, 1993]	5	unequal	AC=AT=CG=GT;AG;GT	010020
TPM1	=K81 [Kimura, 1981]	2	equal	AC=GT;AG=CT;AT=CG	012210
TPM1uf	[Kimura, 1981]	5	unequal	AC=GT;AG=CT;AT=CG	012210
TPM2		2	equal	AC=AT;CG=GT;AG=CT	010212
TPM2uf		5	unequal	AC=AT;CG=GT;AG=CT	010212
TPM3		2	equal	AC=AT;AG=GT;AG=CT	012012
TPM3uf		5	unequal	AC=CG;AT=GT;AG=CT	012012
TIM1	[Posada, 2003]	3	equal	AC=GT;AT=CG;AG;CT	012230
TIM1uf	[Posada, 2003]	6	unequal	AC=GT;AT=CG;AG;CT	012230
TIM2		3	equal	AC=AT;CG=GT;AG;CT	010232
TIM2uf		6	unequal	AC=AT;CG=GT;AG;CT	010232
TIM3		3	equal	AC=CG;AT=GT;AG;CT	012032
TIM3uf		6	unequal	AC=CG;AT=GT;AG;CT	012032
TVMef	[Posada, 2003]	4	equal	AC;CG;AT;GT;AG=CT	012314
TVM	[Posada, 2003]	7	unequal	AC;CG;AT;GT;AG=CT	012314
SYM	[Zharkikh, 1994]	5	equal	AC;CG;AT;GT;AG;CT	012345
GTR	=REV [Tavaré, 1986]	8	unequal	AC;CG;AT;GT;AG;CT	012345

7.2 Sequential Likelihood Ratio Tests (sLRT)

In traditional statistical theory, a widely accepted statistic for testing the goodness of fit of models is the likelihood ratio test (LRT):

$$LRT = 2(l_1 - l_0)$$

where l_1 is the maximum likelihood under the more parameter-rich, complex model (alternative hypothesis) and l_0 is the maximum likelihood under the less parameter-rich simple model (null hypothesis). When the models compared are nested (the null hypothesis is a special case of the alternative hypothesis) and the null hypothesis is correct, the LRT statistic is asymptotically distributed as a χ^2 with q degrees of freedom, where q is the difference in number of free parameters between the two models [Goldman, 1993b; Kendall and Stuart, 1979]. Note that, to preserve the nesting of the models, the likelihood scores need to be estimated upon the same tree. When some parameter is fixed at its boundary (p-inv, α), a mixed χ^2 is used instead [Goldman and Whelan, 2000; Ohta, 1992]. The behavior of the χ^2 approximation for the LRT has been investigated with quite a bit of detail [Goldman, 1993a,b; Goldman and Whelan, 2000; Whelan and Goldman, 1999; Yang et al., 1995].

7.3 Hierarchical Likelihood Ratio Tests (hLRT)

Likelihood ratio tests can be carried out sequentially by adding parameters (forward selection) to a simple model (JC), or by removing parameters (backward selection) from a complex model (GTR+I+G) in a specific order or hierarchy (hLRT; see Figure below). The performance of hierarchical LRTs for phylogenetic model selection has been discussed by Posada and Buckley [2004].

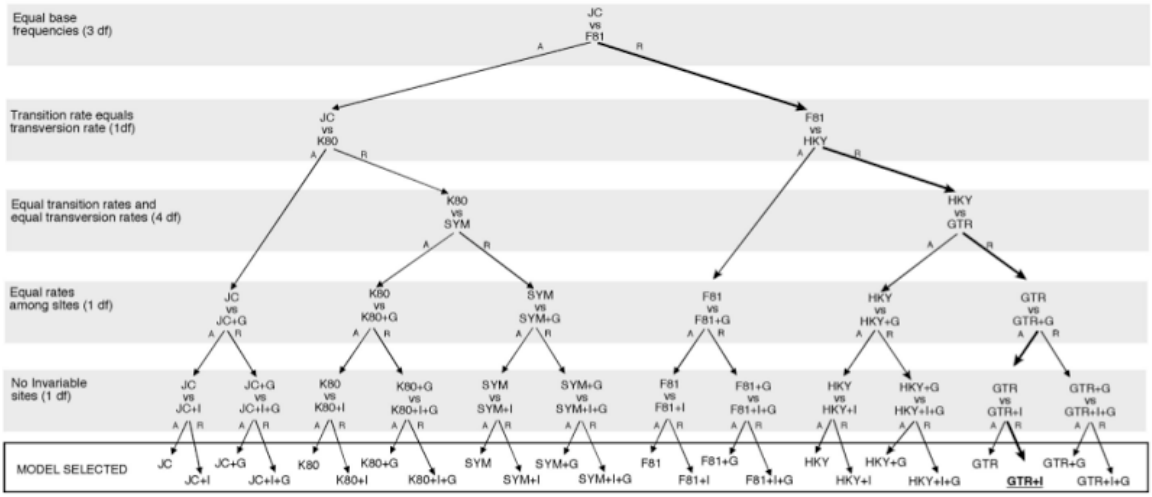


Figure. Example of a particular forward hierarchy of likelihood ratio tests for 24 models. At any level the null hypothesis (model on top) is either accepted (A) or rejected (R). In this example the model selected is GTR+I.

7.4 Dynamical Likelihood Ratio Tests (dLRT)

Alternatively, the order in which parameters are added or removed can be selected automatically. One option to accomplish this is to add the parameter that maximizes a significant gain in likelihood during forward selection, or to add the parameter that minimizes a non-significant loss in likelihood during backward selection [Posada and Crandall, 2001]. In this case, the order of the tests is not specified a priori, but it will depend on the particular data.

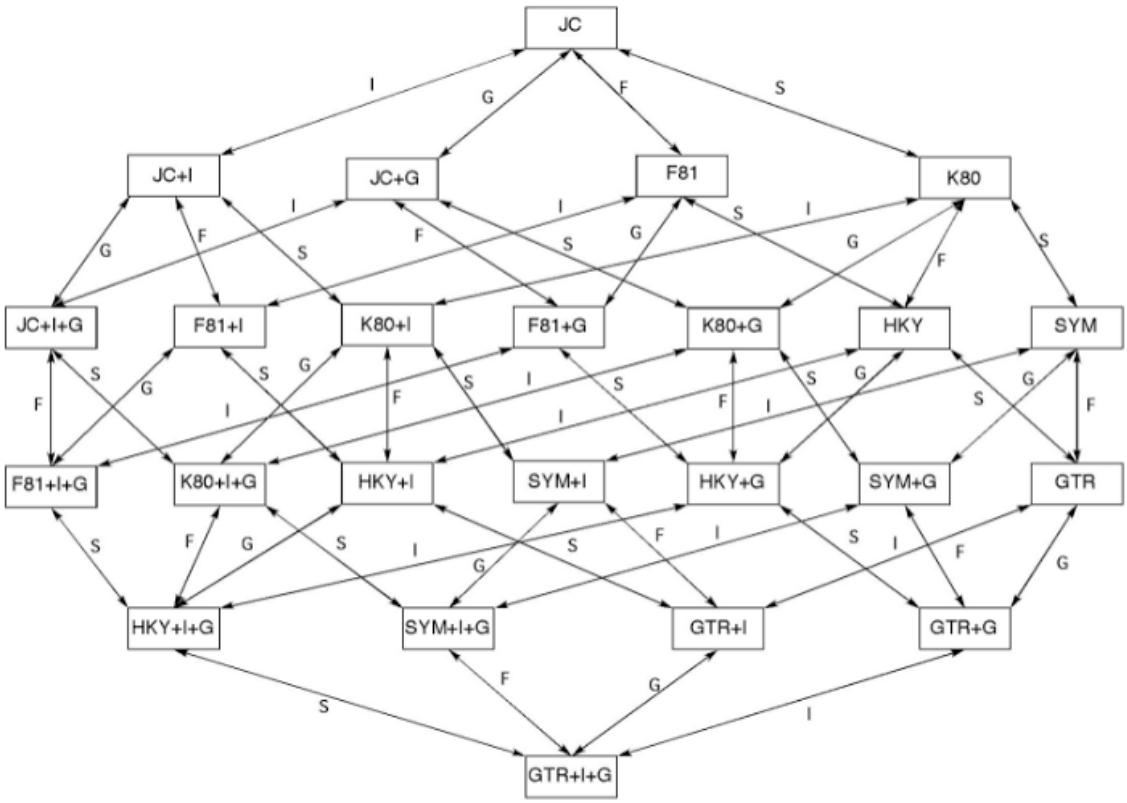


Figure. Dynamical likelihood ratio tests for 24 models. At any level a hypothesis is either accepted (A) or rejected (R). In this example the model selected is GTR+I. Hypotheses tested are: F = base frequencies; S = substitution type; I = proportion of invariable sites; G = gamma rates.

7.5 Information Criteria

7.5.1 Akaike Information Criterion

The Akaike information criterion (AIC, [Akaike, 1974] is an asymptotically unbiased estimator of the Kullback-Leibler information quantity [S. Kullback, 1951]. We can think of the AIC as the amount of information lost when we use a specific model to approximate the real process of molecular evolution. Therefore, the model with the smallest AIC is preferred. The AIC is computed as:

$$AIC = -2l + 2k$$

where l is the maximum log-likelihood value of the data under this model and Ki is the number of free parameters in the model, including branch lengths if they were estimated *de novo*. When sample size (n) is small compared to the number of parameters (say, $\frac{n}{K} < 40$) the use of a second order AIC, AICc [Hurvich and Tsai, 1989; Sugiura, 1978], is recommended:

$$AIC_c = AIC + \frac{(2k(k+1))}{(n-k-1)}$$

The AIC compares several candidate models simultaneously, it can be used to compare both nested and non-nested models, and model-selection uncertainty can be easily quantified using the AIC differences and Akaike weights (see Model uncertainty below). Burnham and Anderson [2003] provide an excellent introduction to the AIC and model selection in general.

7.5.2 Bayesian Information Criterion

An alternative to the use of the AIC is the Bayesian Information Criterion (BIC) [Schwarz, 1978]:

$$BIC = -2l + klog(n)$$

Given equal priors for all competing models, choosing the model with the smallest BIC is equivalent to selecting the model with the maximum posterior probability. Alternatively, Bayes factors for models of molecular evolution can be calculated using reversible jump MCMC [Huelsenbeck et al., 2004]. We can easily use the BIC instead of the AIC to calculate BIC differences or BIC weights.

7.5.3 Performance Based Selection

Minin et al. [2003] developed a novel approach that selects models on the basis of their phylogenetic performance, measured as the expected error on branch lengths estimates weighted by their BIC. Under this decision theoretic framework (DT) the best model is the one with that minimizes the risk function:

$$C_i \approx \sum_{j=1}^{n} ||\hat{B}_i - \hat{B}_j|| \frac{e^{\frac{-BIC_j}{2}}}{\sum_{j=1}^{R}(e^{\frac{-BIC_i}{2}})}$$

where

$$||\hat{B}_i - \hat{B}_j||^2 = \sum_{l=1}^{2t-3} (\hat{B_{il}} - \hat{B_{jl}})^2$$

and where t is the number of taxa. Indeed, simulations suggested that models selected with this criterion result in slightly more accurate branch length estimates than those obtained under models selected by the hLRTs [Abdo et al., 2005; Minin et al., 2003].

7.6 Model Uncertainty

The AIC, Bayesian and DT methods can rank the models, allowing us to assess how confident we are in the model selected. For these measures we could present their differences (Δ). For example, for the ith model, the AIC (BIC, DT) difference is:

$$\Delta_i = AIC_i - min(AIC)$$

where $min(AIC)$ is the smallest AIC value among all candidate models. The AIC differences are easy to interpret and allow a quick comparison and ranking of candidate models. As a rough rule of thumb, models having Δ_i within 1-2 of the best model have substantial support and should receive consideration. Models having Δ_i within 3-7 of the best model have considerably less support, while models with $\Delta_i > 10$ have essentially no support. Very conveniently, we can use these differences to obtain the relative AIC (BIC) weight (w_i) of each model:

$$\omega_i = \frac{e^{\frac{-\Delta_i}{2}}}{\sum_{r=1}^{R}(e^{\frac{-\Delta_r}{2}})}$$

which can be interpreted, from a Bayesian perspective, as the probability that a model is the best approximation to the truth given the data. The weights for every model add to 1, so we can establish an approximate 95% confidence set of models for the best models by summing the weights from largest to smallest from largest to smallest until the sum is 0.95 [Burnham and Anderson, 1998, 2003]. This interval can also be set up stochastically (see above "Model selection and averaging"). Note that this equation will not work for the DT (see the DT explanation on "Model selection and averaging").

7.7 Model Averaging

Often there is some uncertainty in selecting the best candidate model. In such cases, or just one when does not want to rely on a single model, inferences can be drawn from all models (or an optimal subset) simultaneously. This is known as model averaging or multimodel inference. See Posada and Buckley [2004] and references therein for an explanation of application of these techniques in the context of phylogenetics.

Within the AIC or Bayesian frameworks, it is straightforward to obtain a model-averaged estimate of any parameter [Burnham and Anderson, 2003; Hoeting et al., 1999; Madigan and Raftery, 1994; Posada, 2003; Raftery, 1996; Wasserman, 2000]. For example, a model-averaged estimate of the substitution rate between adenine and cytosine using the Akaike weights for R candidate models would be:

$$\widehat{\overline{\phi_{A-C}}} = \frac{\sum_{r=1}^{R} \omega_i I_{\phi_{A-C}}(M_i)\phi_{A-C_i}}{\omega_+(\phi_{A-C})}$$

where

$$\omega_+(\phi_{A-C}) = \sum_{i=1}^{R} \omega_i I_{\phi_{A-C}}(M_i)$$

and

$$I_{\phi_{A-C}}(M_i) = \begin{cases} 1 & \phi_{A-C} \text{ is in model } M_i \\ 0 & \text{otherwise} \end{cases}$$

Note that need to be careful when interpreting the relative importance of parameters. When the number of candidate models is less than the number of possible combinations of parameters, the presence-absence of some pairs of parameters can be correlated, and so their relative importances.

7.8 Model Averaged Phylogeny

Indeed, the averaged parameter could be the topology itself, so we could construct a model-averaged estimate of phylogeny. For example, one could estimate a ML tree for all models (or a best subset) and with those one could build a weighted consensus tree using the corresponding Akaike weights. See Posada and Buckley [2004] for a practical example.

7.9 Parameter Importance

It is possible to estimate the relative importance of any parameter by summing the weights across all models that include the parameters we are interested in. For example, the relative importance of the substitution rate between adenine and cytosine across all candidate models is simply the denominator above, $\omega_+(\phi_{A-C})$

References

Z. Abdo, V.N. Minin, P. Joyce, and J. Sullivan. Accounting for uncertainty in the tree topology has little effect on the decision-theoretic approach to model selection in phylogeny estimation. *Molecular Biology and Evolution*, 22:691–703, 2005.

H. Akaike. A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, 19:716–723, 1974.

K.P. Burnham and D.R. Anderson. Model selection and inference: a practical information-theoretic approach. *Springer-Verlag, New York, NY*, 1998.

K.P. Burnham and D.R. Anderson. Model selection and multimodel inference: a practical information-theoretic approach. *Springer-Verlag, New York, NY*, 2003.

J. Felsenstein. Evolutionary trees from dna sequences: A maximum likelihood approach. *Journal of Molecular Evolution*, 17:368–376, 1981.

J. Felsenstein. Phylogenies from molecular sequences: inference and reliability. *Annual Review of Genetics*, 22: 521–565, 1988.

N. Goldman. Simple diagnostic statistical test of models of dna substitution. *Journal of Molecular Evolution*, 37: 650–661, 1993a.

N. Goldman. Statistical tests of models of dna substitution. *Journal of Molecular Evolution*, 36:182–198, 1993b.

N. Goldman and S. Whelan. Statistical tests of gamma-distributed rate heterogeneity in models of sequence evolution in phylogenetics. *Molecular Biology and Evolution*, 17:975–978, 2000.

M. Hasegawa, K. Kishino, and T. Yano. Dating the human-ape splitting by a molecular clock of mitochondrial dna. *Journal of Molecular Evolution*, 22:160–174, 1985.

J.A. Hoeting, D. Madigan, and A.E. Raftery. Bayesian model averaging: A tutorial. *Statistical Science*, 14: 382–417, 1999.

J. Huelsenbeck, B. Larget, and M.E. Alfaro. Bayesian phylogenetic model selection using reversible jump markov chain monte carlo. *Molecular Biology and Evolution*, 21:1123–1133, 2004.

C.M. Hurvich and C.L. Tsai. Regression and time series model selection in small samples. *Biometrika*, 76: 297–307, 1989.

J.B. Johnson and K.S. Omland. Model selection in ecology and evolution. *Trends in Ecology and Evolution*, 19: 101–108, 2003.

T.H. Jukes and C.R. Cantor. Evolution of protein molecules. *Academic Press, New York, NY*, pages 21–132, 1969.

M. Kendall and A. Stuart. The advanced theory of statistics. *Charles Griffin, London*, 1979.

M. Kimura. A simple method for estimating evolutionary rate of base substitutions through comparative studies of nucleotide sequences. *Journal of Molecular Evolution*, 16:111–120, 1980.

M. Kimura. Estimation of evolutionary distances between homologous nucleotide sequences. *Proceedings of the National Academy of Sciences, U.S.A*, 78:454–458, 1981.

D.M. Madigan and A.E. Raftery. Model selection and accounting for model uncertainty in graphical models using occam's window. *Journal of the American Statistical Association*, 59:1335–1346, 1994.

V. Minin, Z. Abdo, and J. Sullivan P. Joyce. Performance-based selection of likelihood models for phylogeny estimation. *Systematic Biology*, 52:674–683, 2003.

T. Ohta. Theoretical study of near neutrality. ii. effect of subdivided population structure with local extinction and recolonization. *Genetics*, pages 917–923, 1992.

D. Posada. Using modeltest and paup to select a model of nucleotide substitution. pages 6.5.1–6.5.14, 2003.

D. Posada and T.R. Buckley. Model selection and model averaging in phylogenetics: advantages of akaike information criterion and bayesian approaches over likelihood ratio tests. *Systematic Biology*, 53:793–808, 2004.

D. Posada and K. Crandall. Selecting the best-fit model of nucleotide substitution. *Systematic Biology*, 50: 580–601, 2001.

A.E. Raftery. Hypothesis testing and model selection. *Markov chain Monte Carlo in practice. Chapman and Hall, London*, pages 163–187, 1996.

R.A. Leibler S. Kullback. On information and sufficiency. *Annals of Mathematical Statistics*, 22:79–86, 1951.

G. Schwarz. Estimating the dimension of a model. *The Annals of Statistics*, 6:461–464, 1978.

N. Sugiura. Further analysis of the data by akaike's information criterion and the finite corrections. *Communications in Statistics–Theory and Methods*, A7:13–26, 1978.

J. Sullivan and P. Joyce. Model selection in phylogenetics. *Annual Review of Ecology, Evolution and Systematics*, 36:445–466, 2005.

K. Tamura and M. Nei. Estimation of the number of nucleotide substitutions in the control region of mitochondrial dna in humans and chimpanzees. *Molecular Biology and Evolution*, 10:512–526, 1993.

S. Tavaré. Some probabilistic and statistical problems in the analysis of dna sequences. *Some mathematical questions in biology - DNA sequence analysis. Amer. Math. Soc., Providence, RI*, pages 57–86, 1986.

L. Wasserman. Bayesian model selection and model averaging. *Journal of Mathematical Psychology 44:92-107*, 44:92–107, 2000.

S. Whelan and N. Goldman. Distributions of statistics used for the comparison of models of sequence evolution in phylogenetics. *Molecular Biology and Evolution*, 16:1292–1299, 1999.

Z. Yang, N. Goldman, and A.Friday. Maximum likelihood trees from dna sequences: a peculiar statistical estimation problem. *Systematic Biology*, 44:384–399, 1995.

A. Zharkikh. Estimation of evolutionary distances between nucleotide sequences. *Journal of Molecular Evolution*, 39:315–329, 1994.