modeltest[ng]

manual v1.0.0

Diego Darriba, David Posada, Alexandros Stamatakis, Tomas Flouris

September 22, 2017

Contents

1 Overview

ModelTest-NG is a tool to carry out statistical selection of best-fit models of nucleotide substitution or amino acid replacement. 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. *ModelTest-NG* gathers features of *jModelTest 2* [Darriba *et al.*, 2012] and *ProtTest 3* [Darriba *et al.*, 2011].

1.1 Disclaimer

1.2 Download

ModelTest-NG is open source under GNU GPL. It is distributed via github repository: `https://github.com/ddarriba/modeltest` where you can always download the most up to date version. Make sure to watch the github repository to remain up to date regarding code changes. Version numbers follow the notation **x.y.z** where **x** changes with major code reorganizations, **y** changes when new features are added and **z** changes with bug fixes.

Please note that unreleased commits in the trunk may be in testing status.

1.3 Getting help

ModelTest-NG support is provided via the jModelTest discussion group at: `http://groups.google.com/group/jmodeltest`. Note that Google groups have a search function! The answer might also be in this manual.

Also, check the wiki page for up-to-date Frequently Asked Questions: `https://github.com/ddarriba/modeltest/wiki`.

If you believe that you found a bug, or you would like to make a suggestion, feel free to open a ticket at `https://github.com/ddarriba/modeltest/issues`. For bug reports, please provide as many details as possible in order to make it reproducible easily. Please do not open tickets for questions, use the discussion group instead.

2 Getting Started

ModelTest-NG provides graphical and a command console interfaces.

2.1 Compilation

Sources can be compiled for every major Operating System, including Linux, Windows, and Mac OS X. For convenience, with each release you will find binaries for each of these systems. Nonetheless, it might happen that for certain distributions only some of theme are available, for example if the realease fixes a bug affecting one single OS.

This tool is distributed under GPL v3 license. The source code is freely available at github repository: https://github.com/ddarriba/modeltest.

If you clone the repository, make sure to clone also libpll and pll-modules submodules with '–recursive' flag, or call 'git submodule init –recursive' afterwards.

For most users, download the latest release and call the build script, *build.sh*. It should work correctly and (by default) compile all dependencies and place the final binaries under 'build' directory.

- Run *build.sh* with no arguments for compiling all dependencies and modeltest in 3 flavors: *modeltest-ng* (command console execution), *modeltest-mpi* (MPI version), and *modeltest-gui* (graphical user interface). Note that you need QT 4 or 5 for compiling the GUI version.
- Run *build.sh clean* for removing all object and output files.
- Run *build.sh dist* for building a distribution package.

You can change the behaviour and the target directories in the static configuration section, at the beggining of the build script.

```
# Static configuration
build_pll=yes             # build PLL
build_modules=yes         # build PLL modules library
build_gui=yes             # build modeltest-gui
dir_base=${PWD}           # base directory
prefix=${dir_base}/build # output directory for modeltest
qmake_bin=qmake           # qmake binary (for GUI)
```

property	default	description
build_pll	yes	Build the pll dependency
build_modules	yes	Build the pll modules dependency
build_gui	yes	Build the graphical user interface
dir_base	current directory	Base directory
prefix	*dir_base*/build	Target directory
qmake_bin	qmake	qmake binary

Note: QT utility, *qmake* can be also an absolute/relative path, e.g., *qmake_bin=/usr/local/bin/qmake*. For example, I used a custom distribution for OSX and the configuration line looks like this:

```
qmake_bin=/usr/local/Cellar/qt/5.9.1/bin/qmake
```

If everything went fine, the output should look like this:

```
Running install script for Linux
QMake version 3.0
Using Qt version 5.5.1 in /usr/lib/x86_64-linux-gnu
...configuration:
```

```
    prefix:         /home/diego/Repositories/phylogenetics/modeltest-dev/build
    build:          /home/diego/Repositories/phylogenetics/modeltest-dev/build
    pll:            [yes] /home/diego/Repositories/phylogenetics/modeltest-dev/libs/pll-modules/libs/
      libpll
    modules:        [yes] /home/diego/Repositories/phylogenetics/modeltest-dev/libs/pll-modules
       include:     /home/diego/Repositories/phylogenetics/modeltest-dev/build/include/libpll
       lib:         /home/diego/Repositories/phylogenetics/modeltest-dev/build/lib

... writing log to /home/diego/Repositories/phylogenetics/modeltest-dev/build.log
... build pll
    ... building pll
... pll OK!
/home/diego/Repositories/phylogenetics/modeltest-dev
... build modules
    ... building modules
... modules OK!
/home/diego/Repositories/phylogenetics/modeltest-dev
... build modeltest
Project MESSAGE: link to pll local
Project MESSAGE: pll dynamic link
... modeltest GUI OK!
```

The console output contains only trace messages. The full output is redirected to *build.log*, so please check it if there are errors building *ModelTest-NG* .

2.2 Example run

1. Execute the following command line:

```
$ ./modeltest-ng -i example-data/dna/tiny.fas -h uigf -f ef
```

   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.

   See Section 4 for information about supported arguments.

2. This will generate the following output:

   (a) Header:

```
                          _      _ _            _      _   _  -----
                         | |    | | |          | |    | \ | |/ ____|
 _ __ ___   ___   __| | ___| | |_ ___  ___| |_   |  \| | |  __
| '_ ' _ \ / _ \ / _' |/ _ \ | __/ _ \/ __| __|  | . ' | | |_ |
| | | | | | (_) | (_| |  __/ | ||  __/\__ \ |_   | |\  | |__| |
|_| |_| |_|\___/ \__,_|\___|_|\__\___||___/\__|  |_| \_|\_____|
_____________________________________________________________________

modeltest 0.1.0
Copyright (C) 2017 Diego Darriba, David Posada, Alexandros Stamatakis
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

Written by Diego Darriba.
_____________________________________________________________________
```

   (b) System and configuration details:

```
Physical cores: 2
Logical cores:  4
Memory:         3.57GB
Extensions:     AVX
```

   (c) Execution options:

```
Input data:
  MSA:          example-data/dna/aP6.fas
  Tree:         Maximum parsimony
    file:       -
  #taxa:        6
  #sites:       631
  #patterns:    28

Output:
  Log:              test.log
  Starting tree:    test.tree
  Results:          test.out

Selection options:
  # dna schemes:       11
  # dna models:        88
  include model parameters:
    Uniform:          true
    p-inv (+I):       true
    gamma (+G):       true
    both (+I+G):      true
    fixed freqs:      true
    estimated freqs:  true
    #categories:      4
  asc bias:            none
  epsilon (opt):       0.01
  epsilon (par):       0.01

Additional options:
  verbosity:         very low
  threads:           1/2
  RNG seed:          12345
  subtree repeats:   enabled
--------------------------------------------------------------------------------
modeltest-ng was called as follows:
>> src/modeltest-ng -i example-data/dna/aP6.fas -h uifg -f fe -o test
```

(d) Real time optimization results (progress):

```
Partition 1/1

    ID      MODEL      Time      Elapsed          LnL        Alpha    P-inv
  1/88      JC         0h:00:00  0h:00:00   -1115.1193        -        -
  2/88      JC+I       0h:00:00  0h:00:00   -1103.3444        -      0.9082
  3/88      JC+G       0h:00:00  0h:00:00   -1106.6136     0.0200      -
  4/88      JC+I+G     0h:00:00  0h:00:00   -1103.6235     1.1674   0.8542
  5/88      F81        0h:00:00  0h:00:00   -1065.0339        -        -
  6/88      F81+I      0h:00:00  0h:00:00   -1053.6319        -      0.9032
  7/88      F81+G      0h:00:00  0h:00:00   -1056.6126     0.0200      -
  8/88      F81+I+G    0h:00:00  0h:00:00   -1053.8953     1.1494   0.8460

...

 85/88      GTR        0h:00:00  0h:00:01   -1063.2358        -        -
 86/88      GTR+I      0h:00:00  0h:00:01   -1051.9056        -      0.9001
 87/88      GTR+G      0h:00:00  0h:00:01   -1054.7872     0.0200      -
 88/88      GTR+I+G    0h:00:00  0h:00:01   -1052.1689     1.1396   0.8417
    ID      MODEL      Time      Elapsed          LnL        Alpha    P-inv

Computation of likelihood scores completed. It took 0h:00:01
```

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

BIC	model	K	lnL	score	delta	weight
1	F81+I	4	-1053.6319	2191.0788	0.0000	0.8565
2	HKY+I	5	-1053.1557	2196.5737	5.4949	0.0549
3	F81+G	4	-1056.6126	2197.0401	5.9613	0.0435
4	F81+I+G	5	-1053.8953	2198.0529	6.9741	0.0262
5	TrN+I	6	-1052.6019	2201.9134	10.8346	0.0038
6	TPM2uf+I	6	-1052.6600	2202.0296	10.9507	0.0036
7	HKY+G	5	-1056.0996	2202.4615	11.3827	0.0029

```
     8   TPM3uf+I          6      −1052.9534      2202.6164      11.5376      0.0027
     9   TPM1uf+I          6      −1053.0742      2202.8579      11.7791      0.0024
    10   HKY+I+G           6      −1053.4340      2203.5777      12.4988      0.0017
-------------------------------------------------------------------------------------
Best model according to BIC
---------------------------
Model:                  F81+I
lnL:                    −1053.6319
Frequencies:            0.4253 0.1506 0.2010 0.2232
Subst. Rates:           1.0000 1.0000 1.0000 1.0000 1.0000 1.0000
Inv. sites prop:        0.9032
Gamma shape:            −
Score:                  2191.0788
Weight:                 0.8565
---------------------------
Parameter importances
---------------------------
P.Inv:                  0.9244
Gamma:                  0.0471
Gamma−Inv:              0.0282
Frequencies:            1.0000
---------------------------
Model averaged estimates
---------------------------
P.Inv:                  0.9031
Alpha:                  0.0200
Alpha−P.Inv:            1.1502
P.Inv−Alpha:            0.8459
Frequencies:            0.4253 0.1506 0.2010 0.2232

Commands:
  > phyml  −i example−data/dna/aP6.fas −m 000000 −f m −v e −a 0 −c 1 −o tlr
  > raxmlHPC−SSE3 −s example−data/dna/aP6.fas −c 1 −m GTRCATIX −−JC69 −n EXEC_NAME −p
    PARSIMONY_SEED
  > paup −s example−data/dna/aP6.fas
  > iqtree −s example−data/dna/aP6.fas −m F81+I
```

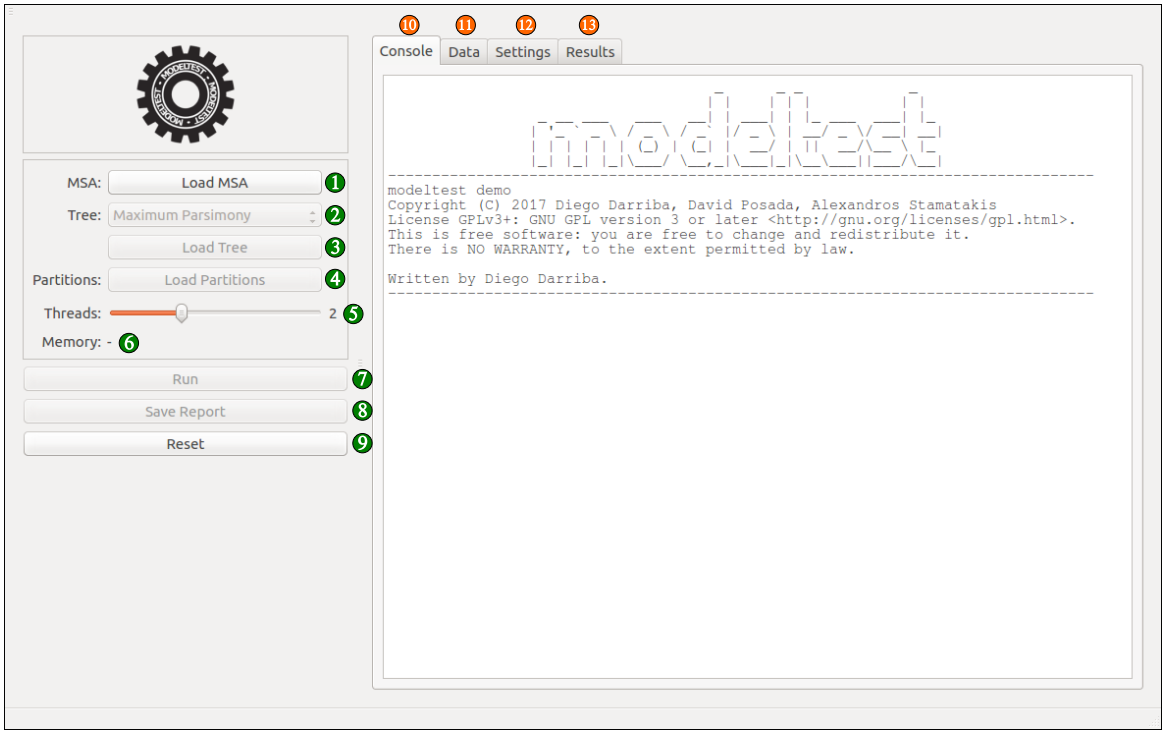
(f) Consensus tree of the optimized phylogenies using the criterion weights (only for **ML topologies**):

```
There are 2 different topologies
Topologies written to output.topos

topo_id    models_count    bic_support    aic_support    aicc_support
-------------------------------------------------------------------------
      1              37        0.95897        0.66064         0.66964
      2              51        0.04103        0.33936         0.33036

extended majority−rule consensus: ((P4,(P6,P1)[1.00000])[0.95897],P5,(P2,P3)[1.00000]);
strict consensus: ((P6,P1)[1.00000],P4,P5,(P2,P3)[1.00000]);
```

3 Graphical User Interface

3.1 Launching the Graphical User Interface

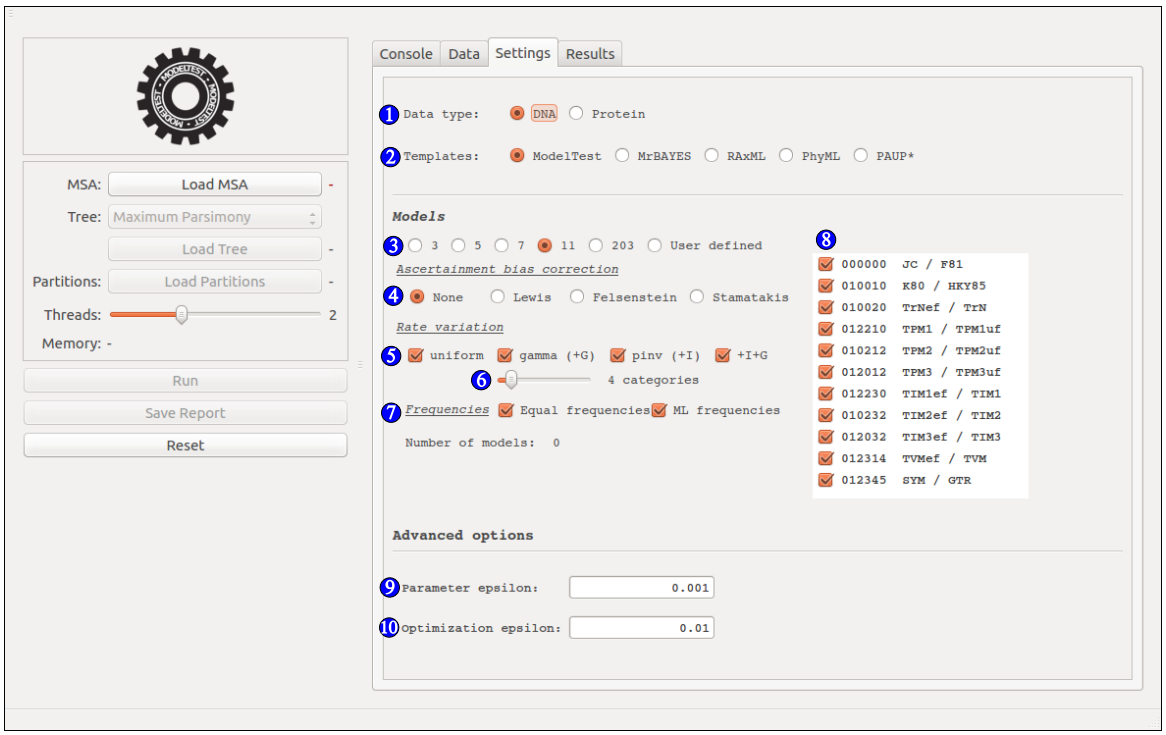
Running *modeltest-gui* with no arguments launches the graphical interface. The following window will show on the screen:



1	Load an MSA file in PHYLIP or FASTA format
2	Select the phylogenetic tree for each model
3	Load a fixed or starting tree in NEWICK format (optional)
4	Load a partitioning scheme file in RAxML format (optional)
5	Select the number of concurrent threads to use
6	Displays the estimated amount of memory needed as a function of the MSA size and the number of threads
7	Start model selection process
8	Save the results report in a file
9	Reset the interface
10	Pane containing the main output console
11	Pane containing data description
12	Pane containing the model selection configuration
13	Pane containing the model selection results

3.2 Custom settings

The settings tab (12) allows to change the model optimization settings. Although the default settings are the most commonly used, you might want to use different ones for your own purposes.



1 Data type (DNA or amino acids)
2 Use only models available in a particular phylogenetic inference tool
3 Use *a priori* defined subset of substitution schemes
4 Correct models for ascertainment bias
5 Include models of rate variation among sites
6 Select the number of discrete rate categories for Gamma model of rate variation
7 Include equal/model-defined or ML/empirical frequencies
8 Select individual candidate models
9 Tolerance for single parameter optimization
10 Global tolerance for model optimization

3.3 Example

If you want to start running a small example, press Ctrl+O in the main window. Select a MSA file from 'example-data/nucleic' or 'example-data/proteic' in the dialog, either in FASTA or PHYLIP format. Press Ctrl+T and select the corresponding tree file in the dialog, in NEWICK format. Press Ctrl+R and enjoy the execution.

4 Command Line Arguments

4.1 Overview

4.1.1 Main Arguments

`-d`	`--datatype`	**nt,aa**	Data type is 'nt' for nucleotide (default), 'aa' for amino-acid sequences.
`-i`	`--input`	*filename*	Input MSA file in FASTA or sequential PHYLIP format. Check section 5.1
`-t`	`--topology`	*topology_type*.	Check section 5.3
		ml	maximum likelihood
		mp	maximum parsimony (default)
		fixed-ml-jc	fixed maximum likelihood (JC)
		fixed-ml-gtr	fixed maximum likelihood (GTR)
		random	random generated tree
		user	fixed user defined (requires -u argument)
`-u`	`--utree`	*filename*	User-defined tree in NEWICK format. Check section 5.3
`-q`	`--partitions`	*filename*	Partitions filename in RAxML format. Check section 5.4
`-o`	`--output`	*filename*	Pipes the output into a file
`-p`	`--processes`	*number_of_threads*	Number of concurrent threads
`-r`	`--rngseed`	*seed*	Sets the seed for the random number generator

4.1.2 Candidate Models

`-a`	`--asc-bias`	algorithm[:values]	Includes ascertainment bias correction. Check section 5.5 for more details **lewis**: Lewis (2001) **felsenstein**: Felsenstein (requires number of invariant sites) **stamatakis**: Leach et al. (2015) (requires invariant sites composition)
`-f`	`--frequencies`	[ef]	Sets the candidate models frequencies **e**: Estimated - maximum likelihood (DNA) / empirical (AA) **f**: Fixed - equal (DNA) / model defined (AA)
`-h`	`--model-het`	[uigf]	Sets the candidate models rate heterogeneity **u**: Uniform **i**: Proportion of invariant sites (+I) **g**: Discrite Gamma rate categories (+G) **f**: Both +I and +G (+I+G)
`-m`	`--models`	*list*	Sets the candidate model matrices separated by commas
		dna:	JC HKY TrN TPM1 TPM2 TPM3 TIM1 TIM2 TIM3 TVM GTR
		protein:	DAYHOFF LG DCMUT JTT MTREV WAG RTREV CPREV VT BLOSUM62 MTMAM MTART MTZOA PMB HIVB HIVW JTTDCMUT FLU STMTREV
`-s`	`--schemes`	*number_of_schemes*	Number of DNA substitution schemes. **3**: JC, HKY, GTR **5**: JC, HKY, TrN, TPM1, GTR **7**: JC, HKY, TrN, TPM1, TIM1, TVM, GTR **11**: All models defined in Sec 5.2 **203**: All possible GTR submatrices
`-T`	`--template`	*tool*	Sets candidate models according to a specified tool
		raxml	RAxML (DNA 3 schemes / AA full search)
		phyml	PhyML (DNA full search / 14 AA matrices)
		mrbayes	MrBayes (DNA 3 schemes / 8 AA matrices)
		paup	PAUP* (DNA full search / AA full search)

4.1.3 Other options

	`--eps`	*epsilon_value*	Sets the model optimization epsilon
	`--tol`	*tolerance_value*	Sets the parameter optimization tolerance
	`--smooth-frequencies`		Forces frequencies smoothing
`-H`	`--no-compress`		Disables pattern compression. *ModelTest-NG* ignores if there are missing states
`-v`	`--verbose`		Run in verbose mode
	`--help`		Display this help message and exit
	`--version`		Output version information and exit

5 Model Optimization Settings

5.1 Input data

The main and only required argument is the multiple sequence alignment file ($-i$ argument). *ModelTest-NG* supports PHYLIP and FASTA format. All sequences must be alignned and have thus have the same sequence length.

PHYLIP format starts with a header line containing 2 integer values corresponding to the number of sequences and the sequence length. The following lines are the individual taxa followed by the corresponding sequence. Taxon names and sequences must *not* contain whitespaces. If that is the case in your alignment, please remove or replace every white space with any arbitrary character, such for example an underscore.

Please note that at this moment *ModelTest-NG* does not support interleaved PHYLIP format.

```
TAXA_COUNT SEQ_LENGTH
TAXON_NAME_1 SEQUENCE_1
TAXON_NAME_2 SEQUENCE_2
TAXON_NAME_3 SEQUENCE_3
...
TAXON_NAME_N SEQUENCE_N
```

Example:

```
5 20
taxon1 acgctatcgcgatcgatagc
taxon2 aaactagggcgatcgatagg
taxon3 acactatcg---tcgatagg
taxon4 acgctatcg---ccgatagg
taxon5 acgctaacgcgaacgttatc
```

FASTA format does not contain any header, and it is formatted as a list of the sequences, each of them covering 2 lines: the taxon name, and the sequence.

```
>TAXON_NAME_1
SEQUENCE_1
>TAXON_NAME_2
SEQUENCE_2
>TAXON_NAME_3
SEQUENCE_3
...
>TAXON_NAME_N
SEQUENCE_N
```

The example below is analogous to the previous example in PHYLIP format:

```
>taxon1
acgctatcgcgatcgatagc
>taxon2
aaactagggcgatcgatagg
>taxon3
acactatcg---tcgatagg
>taxon4
acgctatcg---ccgatagg
>taxon5
acgctaacgcgaacgttatc
```

5.2 Models of evolution

ModelTest-NG implementes all 203 types of time-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:

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

ModelTest-NG includes the empirical amino acid matrices described in the table below. If you expect a very long runtime according to the size of your data, it is recommended to select *a priori* a sensible set of candidate matrices instead of evaluating all the available ones (e.g., discarding those matrices estimated from different data).

Model	Description
Dayhoff	General matrix [Dayhoff and Schwartz, 1978]
JTT	General matrix [Jones *et al.*, 1992]
DCMut/JTT-DCMut	Revised Dayhoff and JTT matrices [Kosiol and Goldman, 2005]
WAG	General matrix [Whelan and Goldman, 2001]
VT	General matrix [Müller and Vingron, 2000]
cpREV	Chloroplast matrix [Adachi *et al.*, 2000]
rtREV	Retrovirus [Dimmic *et al.*, 2002]
stmtREV	Streptophyte mitochondrial land plants [Liu *et al.*, 2014]
mtArt	Mitochondrial Arthropoda [Abascal *et al.*, 2007]
mtMam	Mitochondrial Mammals [Yang and Nielsen, 1998]
mtREV	Mitochondrial Verterbrate [Adachi and Hasegawa, 1996]
mtZoa	Mitochondrial Metazoa (Animals) [Rota-Stabelli *et al.*, 2009]
HIVb/HIVw	HIV matrices [Nickle *et al.*, 2007]
LG	General matrix [Le and Gascuel, 2008]
Blosum62	BLOcks SUbstitution Matrix [Henikoff and Henikoff, 1992]
PMB	Revised Blosum matrix [Veerassamy *et al.*, 2003]
FLU	Influenza virus [Dang *et al.*, 2010]
LG4M	4-matrix mixture model with discrete Γ rates [Le *et al.*, 2012]
LG4X	4-matrix mixture model with free rates [Le *et al.*, 2012]

5.3 Topology type

By default, *ModelTest-NG* optimizes each single model using a fixed Maximum-Parsimony topology with Maximum-Likelihood optimized branch lengths. However, it allows other tree optimization techniques. The topology type can be selected with $-t$ argument and it accepts the following values:

- **ml:** Optimize topology and branch lengths for each model
- **fixed-ml-jc:** Build a ML topology with Jukes-Cantor model and fixes it for every other.
- **fixed-ml-gtr:** Build a ML topology with GTR model and fixes it for every other.
- **random:** Use a fixed randomly generated tree.
- **user:** Use fixed user-defined topology

In addition to that, you can set a custom tree topology using $-u$ argument, followed by a file containing the tree in NEWICK format. *ModelTest-NG* works only with unrooted topologies, so if the user-defined tree is rooted, it will be automatically unrooted. This argument is mandatory if the tree type was set to *u*ser, and optional for ML trees. In the latter case, the custom-defined tree is used as starting point for the ML optimization, while otherwise *ModelTest-NG* uses a MP tree.

A *r*andom tree topology can be interesting if one wants to measure how sensitive is the model selection process to the tree topology. If you want to test several different random trees, do not forget to use different RNG seeds ($-r$ argument).

5.4 Partitioning scheme

ModelTest-NG is able to select individual models of evolution for each partition defined on the data set ($-q$ argument). The partitioning scheme used may be defined in a file using RAxML-like format, where each partition is defined by one line in the file as follows:

```
DATA_TYPE, PARTITION_NAME = PARTITION_SITES
```

Where:

- **DATA_TYPE** can be *D*NA or *P*ROTEIN
- **PARTITION_NAME** is an arbitrary name for each partition
- **PARTITION_SITES** is the subset of sites that belong to the partition. They can be contiguous (e.g.,1-1000), or defined in several sections (e.g., $1-1000, 2500-3000$). Additionally, one can specify a stride. For example, a partition covering all first codon positions in the first 1,000 sites is defined as $1-1000\backslash 3$, second codon position is $2-1000\backslash 3$, and third $3-1000\backslash 3$. Second and third codon positions together would be $2-1000\backslash 3, 3-1000\backslash 3$.

For example:

```
DNA, GENE1 = 1-800
DNA, GENE2_1 = 1701-2400\3
DNA, GENE2_2 = 1702-2400\3
DNA, GENE2_3 = 1703-2400\3
DNA, GENE3 = 801-1700,2401-2500
```

Partitions do not need to cover all sites in the MSA. Every site which does not belong to any partition is just ignored. Also, there must not be overlapping partitions (i.e., it is not allowed a site to belong to more than one partition).

5.5 Ascertainment Bias Correction

ModelTest-NG incorporates 3 algorithms for including ascertainment bias correction in the candidate models.

Let c be the sum of likelihoods (**not** log-likelihoods) of the 'dummy', or virtual invariant sites containing each of the states (eq. 1):, n is the number of sites, s is the number of states, ω is the number of invariant sites, and ω_i is the number of invariant sites for state i.

$$c = \sum_{i}^{s} L(s) \tag{1}$$

- Lewis (Lewis, 2001)

$$ln(L) = \sum_{i}^{n} ln(L_i) - n \cdot ln(1-c) \tag{2}$$

- Felsenstein (Felsenstein, xx)

$$ln(L) = \sum_{i}^{n} ln(L_i) + \omega \cdot ln(c) \tag{3}$$

- Stamatakis (Leaché et al. 2015)

$$ln(L) = \sum_{i}^{n} ln(L_i) + \sum_{j}^{s} \omega_j \cdot ln(L(j)) \tag{4}$$

You can set ascertainment bias correction in *ModelTest-NG* using the *-a* argument: -a algorithm[:values], where *a*lgorithm must be *l*ewis, *f*elsenstein or *s*tamatakis. Additionally, the weights of the dummy sites for Felsenstein's and Stamatakis' algorithms can be set using the *v*alue optional argument. For example:

- Lewis' algorithm (no weights required)

```
$ modeltest -i example-data/dna/aP6.fas -a lewis
```

- Felsenstein's algorithm (sum of dummy sites weights required, values=$w_a + ... + w_t$)

```
$ modeltest -i example-data/dna/aP6.fas -a felsenstein:20
```

- Stamatakis' algorithm (dummy sites weights required, values="w_a, w_c, w_g, w_t")

```
$ modeltest -i example-data/dna/aP6.fas -a stamatakis:10,5,7,15
```

The weights can also be set in the partitions file in a RAxML-like manner, because if the analysis involves several partitions, the dummy sites weights are likely unequal.

There are 2 important conditions for using ascertainment bias correction:

1. The input alignment must *n*ot contain invariant sites.
2. Models with a proportion of invariant sites (i.e., +I and +I+G must be excluded. If -h argument for selecting the rate variation is present and it includes 'g' or 'f', *ModelTest-NG* will complain and stop.

5.6 Frequencies

Nucleotide or amino acid stationary frequencies in a model of evolution can be either (i) defined *a-priori*, using fixed equal or empirical frequencies, or (ii) estimated from the data set at hand, computing the empirical frequencies or estimating ML ones. The latter involve $S-1$ additional degrees of freedom, where S is the number of states (4 for DNA, 20 for protein data).

For nucleotide substitution models, *ModelTest-NG* supports equal (no additional degrees of freedom) and ML frequencies (3 additional degrees of freedom).

For amino acid replacement models, *ModelTest-NG* supports model-defined (no additional degrees of freedom) and empirical frequencies (19 additional degrees of freedom).

With $-f$ argument you can choose whether you want to include models with fixed and/or estimated frequencies using one of both options below. By default, *ModelTest-NG* behaves as including the argument $-f\ ef$.

Arg	Nucleotide	Amino acid
f	fixed equal	fixed model
e	ML estimated	empirical

5.7 Per-site rate heterogeneity

With $-h$ argument you can choose whether you want to include models with per-site rate heterogeneity using one or more options below. By default, *ModelTest-NG* behaves as including the argument $-h\ uigf$.

Arg	Rate heterogeneity model
u	No rate heterogeneity
i	proportion of invariant sites (+I)
g	discrete Gamma rates (+G)
f	both +I and +G together

5.8 Substitution schemes

5.9 Settings templates

In order to use the model of evolution selected by *ModelTest-NG* in other phylogenetic inference tool, you can select one of the settings templates such that you can make sure that the candidate models set contains only models available in specific tools:

- RAxML: JC/F81, K80/HKY and SYM/GTR models, with 4 gamma rate categories and a proportion of invariable sites.
- MrBayes: JC/F81, K80/HKY and SYM/GTR models, with 4 gamma rate categories and a proportion of invariable sites.

5.10 Custom optimization thoroughness

Thoroughness of the optimization process can be fine-tuned using 2 parameters: a local tolerance parameter controls the convergence criteria for optimizing individual parameters, and a global tolerance parameter decides whether to finish individual model optimization based on the log-likelihood score.

6 Common Use Cases

6.1 Loading Checkpointing Files

ModelTest-NG saves a ".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. If a checkpoint file exists for the input MSA, *ModelTest-NG* will check if the current arguments are the same (or compatible) as in the saved search. If not, it will return an error, because that means that the stored models were evaluated under different conditions and the results would be inconsistent. You should then either restart the search with the previous arguments, or remove the ".ckp" file.

References

Abascal, F., Posada, D., and Zardoya, R. (2007). Mtart: a new model of amino acid replacement for arthropoda. *Molecular biology and evolution*, **24**(1), 1–5.

Adachi, J. and Hasegawa, M. (1996). Model of amino acid substitution in proteins encoded by mitochondrial dna. *Journal of molecular evolution*, **42**(4), 459–468.

Adachi, J., Waddell, P. J., Martin, W., and Hasegawa, M. (2000). Plastid genome phylogeny and a model of amino acid substitution for proteins encoded by chloroplast dna. *Journal of Molecular Evolution*, **50**(4), 348–358.

Dang, C. C., Le, Q. S., Gascuel, O., and Le, V. S. (2010). Flu, an amino acid substitution model for influenza proteins. *BMC evolutionary biology*, **10**(1), 99.

Darriba, D., Taboada, G. L., Doallo, R., and Posada, D. (2011). Prottest 3: fast selection of best-fit models of protein evolution. *Bioinformatics*, **27**(8), 1164–1165.

Darriba, D., Taboada, G. L., Doallo, R., and Posada, D. (2012). jmodeltest 2: more models, new heuristics and parallel computing. *Nature methods*, **9**(8), 772–772.

Dayhoff, M. O. and Schwartz, R. M. (1978). A model of evolutionary change in proteins. In *In Atlas of protein sequence and structure*. Citeseer.

Dimmic, M. W., Rest, J. S., Mindell, D. P., and Goldstein, R. A. (2002). rtrev: an amino acid substitution matrix for inference of retrovirus and reverse transcriptase phylogeny. *Journal of molecular evolution*, **55**(1), 65–73.

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

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

Henikoff, S. and Henikoff, J. G. (1992). Amino acid substitution matrices from protein blocks. *Proceedings of the National Academy of Sciences*, **89**(22), 10915–10919.

Jones, D. T., Taylor, W. R., and Thornton, J. M. (1992). The rapid generation of mutation data matrices from protein sequences. *Computer applications in the biosciences: CABIOS*, **8**(3), 275–282.

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

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

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

Kosiol, C. and Goldman, N. (2005). Different versions of the dayhoff rate matrix. *Molecular biology and evolution*, **22**(2), 193–199.

Le, S. Q. and Gascuel, O. (2008). An improved general amino acid replacement matrix. *Molecular biology and evolution*, **25**(7), 1307–1320.

Le, S. Q., Dang, C. C., and Gascuel, O. (2012). Modeling protein evolution with several amino acid replacement matrices depending on site rates. *Molecular biology and evolution*, page mss112.

Liu, Y., Cox, C. J., Wang, W., and Goffinet, B. (2014). Mitochondrial phylogenomics of early land plants: mitigating the effects of saturation, compositional heterogeneity, and codon-usage bias. *Systematic biology*, **63**(6), 862–878.

Müller, T. and Vingron, M. (2000). Modeling amino acid replacement. *Journal of Computational Biology*, **7**(6), 761–776.

Nickle, D. C., Heath, L., Jensen, M. A., Gilbert, P. B., Mullins, J. I., and Pond, S. K. (2007). Hiv-specific probabilistic models of protein evolution. *PLoS One*, **2**(6), e503.

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

Rota-Stabelli, O., Yang, Z., and Telford, M. J. (2009). Mtzoa: a general mitochondrial amino acid substitutions model for animal evolutionary studies. *Molecular phylogenetics and evolution*, **52**(1), 268–272.

Tamura, K. and Nei, M. (1993). 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.

Tavaré, S. (1986). 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.

Veerassamy, S., Smith, A., and Tillier, E. R. (2003). A transition probability model for amino acid substitutions from blocks. *Journal of Computational Biology*, **10**(6), 997–1010.

Whelan, S. and Goldman, N. (2001). A general empirical model of protein evolution derived from multiple protein families using a maximum-likelihood approach. *Molecular biology and evolution*, **18**(5), 691–699.

Yang, Z. and Nielsen, R. (1998). Synonymous and nonsynonymous rate variation in nuclear genes of mammals. *Journal of molecular evolution*, **46**(4), 409–418.

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