JML, version 1.03

Simon Joly

November 24, 2014

Institut de recherche en biologie végétale, Université de Montréal & Montreal Botanical Garden, 4101 Sherbrooke Est, Montréal, Québec H1X 2B2; Email: simon.joly@umontreal.ca

Introduction

JML is a program that implements the statistical method of Joly et al. [1] for detecting introgessed sequences. The method uses posterior predictive checking [2] to test whether the minimum distance between sequences of two species is smaller than expected under a scenario that does not account for hybridization. If the observed distance is smaller than, say, 95% of the simulated valued, then we can conclude that lineage sorting can not explain the data and an hypothesis of hybridization can be accepted.

JML uses as input posterior distributions of species trees, population sizes and divergence times. It can thus fully incorporate the uncertainty involved with these estimates, which can often be important, when testing hypotheses of hybridization. Practicaly, this is done by using the output of the *BEAST [3], a recently developped method that is implemented in BEAST [4]. JML then samples species trees (with branch lengths and population sizes) from the Markov Chain Monte Carlo output and for each species tree, it simulates gene trees, then DNA sequences and finally estimates the minimum distance between sequences for all pairs of species. This will generate posterior predictive distributions that can then be used to estimate the *p*-values of empirical minimum sequences distances between species.

The JML program uses code from other programs to generate the posterior predictive distributions. Gene genealogies are generated using the same code as the MCMCCOAL program [5] by Ziheng Yang and the sequences are simulated using code from the program SEQ-GEN [6], that is in part based on code from the software PAML [7].

Warning

This program has been tested on a few datasets and appears to be behaving correctly. However, there ise no guaranty that is will perform correctly on all dataset. Please, *use with care* and thoroughly inspect the results to make sure they make sense. The program outputs a species tree in MCMCcoal format (newick format with '#' to indicate population sizes) to the buffer. Inspect it to make sure it looks fine. If it doesn't, then you should not trust the results. Please contact me if you find anything that looks anormal.

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., 51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA.

Downloading

The JML software can be downloaded from the following address:
www.plantevolution.org/jml.html.

Version history

1.03

- Minor change in the code to make the program more stable when reading files.
- Now accepts species names longer than 10 characters as the software now uses a relaxed phylip format for sequence files.

1.02

- Minor issue corrected that caused the program to crash after the simulations when no sequence file was given as input. This had no impact on the results though.
- The program gives more information on what it is exactly doing. This is in order to help find which file is problematic when the program crashes.

- The program doesn't prompt for a sequence file anymore... this need to be given in the command line.
- JML outputs the usage with the '-h' flag on the command line.
- Released May 22nd, 2013

1.01

- The program doesn't prompt for values. The values for the burnin, the thinning and the seed can be given in the control file
- Released February 21th, 2012

1.00

- Released October 5th, 2011

Compiling

A JML executable file for Windows OS is located in the 'bin' folder. Binaries for UNIX-based environments need to be compiled. Source files and a makefile are located in the folder 'src'. To compile JML, just go in the folder ./src and type `make`.

Input tree file

JML takes as input tree files obtained with the software *BEAST, which is a slightly modified nexus format. This allows to incorporate uncertainty in the species tree topology as well as in the divergence times and mutational populations sizes. It is possible to input species tree from other program as long as the file is in the same type of nexus format as the *BEAST tree files (look at the examples for a typical file). It is also possible to fix a species tree topology. To do this, one would need to create a tree file in *BEAST format with the same tree repeated a number of times that is equal to the number of simulations to perform. If one wants to vary the population sizes and branch lengths but not the tree topology (such as when using the output of MCMCcoal) then each species tree in the tree file should have a distinct set of branch lengths and population sizes. Unfortunately, there is not tool available to convert an MCMCcoal (or IM) output into a *BEAST nexus tree format.

Running *BEAST

*BEAST is implemented in the program BEAST, available at http://beast.bio.ed.ac.uk.
A tutorial for running the sofware can be downloaded from http://beast-mcmc.googlecode.com/files/STAR-BEAST.zip. **IMPORTANT**: It is important that you choose the

option “piecewise constant” for the population size model (from the ’tree’ tab). Presently, the JML program assumes that this option has been chosen. The *BEAST program outputs a file <your_xml_file_name>.species.trees that will contain the posterior samples for the species tree. This is the file that needs to be input in JML. For greater compatility with the jml software, it is recommended to use species names that are of 7 characters or less (see below).

Ploidy type

When you are preparing your xml file in BEAUTi for the *BEAST run, you are asked to enter a ploidy type for each marker. Persently, the way this value is treated in *BEAST is a bit weird and requires a special attention to be added. It seems that presently, if you have all markers of the same type (say all mitochondrial or all autosomal nuclear), then no scale are given in the xml file. This means that the population sizes output by *BEAST are that of the markers used. If you have markers of more than one type, then generally *BEAST will output the haploid population size. This means that if you have a mitoshondrial and an autosomal marker in an analysis, then the mitochondrial population size will be scaled by a factor of 0.5 relative to the results and that the autosomal nuclear marker will be scaled by a factor of 2 (i.e., the effective population size for the autosomal nuclear marker is twice that output by *BEAST).

I strongly suggest that you verify which ploidy types were used in *BEAST by inspecting the xml file as this can have a dramatic effect on the results obtained in JML. In the xml file, you need to go in the section called “collection of Gene Trees”. It should look likt this:

```
<!-- Collection of Gene Trees -->
<geneTrees id="geneTrees">
    <gtree ploidy="1">
        <treeModel idref="chloroplast.treeModel"/>
    </gtree>
    <gtree ploidy="2.0">
        <treeModel idref="nuclear.treeModel"/>
    </gtree>
</geneTrees>
```

Pay a close attention to the ploidy value given to each marker.

Performing simulations

The executable should be in the same folder as the species tree file (*.species.trees) to perform the simulations. The folder should also contain the JML control file (jml.ctl) as well as the sequence file (optional) if you want to calculate exact probabilities of the minimum observed distances.

JML control file

Some information need to be given to JML to perform the simulations. This information is given in a control file named jml.ctl. The format of the control file is simple: each line contains a command name, followed by an 'equal' sign, and then by the value(s) that go(es) with the command. The following commands should be present in the control file. An example control file (/jml/example/jml.tpi.ctl) is provided with the program.

speciesnames

You need to provide the name of the species present in the species trees provided. Make sure that the name are *identical* to those present in the species tree file. There needs to be at least one white space between each species.

```
speciesnames = homo chimp gorilla
```

IMPORTANT: JML presently use a strict phylip format when writing the sequence files. For this reason, it is strongly recommended that you limit the name of species to a maximum of 7 characters. Presently, long species name could cause the program to crash or give incorrect results. It may also be a good idea to use these names also in *BEAST to make the work flow smoother.

seqinspecies

For each species name provided in the speciesnames command, you need to provide the number of sequences present in the original dataset for each species. It is IMPORTANT to follow the same order as the speciesnames block.

```
seqinspecies = 4 5 3
```

locusrate [DEFAULT = 1]

This gives the relative mutation rate of the locus you want to simulate relative to the one that is incorporated in the *BEAST tree. If the locus you are simulating was included in the *BEAST analysis, the "relative" locusrate of the locus can be found in the beast output files (*.log). You can enter the median or the mean value here. In contrast, if the locus you are simulating was not included in the *BEAST search, then you need to provide a locus rate using another mean (e.g., distances between pair of species). This locus rate is very important to scale the species divergence and the population size appropriately for the locus that will be simulated.

```
locusrate = 0.876
```

heredityscalar [DEFAULT = 1]

As for the locusrate parameter, the heredity scalar is the constant that will be multiplied to the population sizes provided in the species.trees file to obtain the proper effective population size for the simulations. For instance, if the population sizes provided in the *.species.trees file are for autosomal genes and if you want to simulate chloroplast genes for hermaphrodite individuals, then the heredity scalar will be 0.5 as the effective population size of the chloroplast is half that of autosomal genes in hermaphrodite individuals.

```
heredityscalar = 0.5
```

It is very important that you make sure of the scale used for the population size output by *BEAST (see above). In the example given above where the ploidy level of the autosomal nuclear marker was 2 in *BEAST, a heredity scalar of 2 should be used in JML.

seqgencommands

A seq-gen command needs to be provided. This will correspond to the best substitution model for the locus you want to simulate. You should refer to seq-gen for the options available. For example,

```
seqgecommand = -mGTR -f0.2693,0.1671,0.2168,0.3469
 -r1,2.2497,0.2497,0.2497,2.2497,1 -i0 -l716.
```

significancelevel [DEFAULT = 0.1]

If the original sequence file is provided (see below for more information on the format), JML will give all the pairwise sequence distances for which the *p*-value is smaller than this significance level. The default is to report distances that have a *p*-value < 0.1.

```
significancelevel = 0.1
```

burnin [DEFAULT = 10% of the total number of trees]

This allows to discard a certain number of trees from the simulation that were sampled before the chain reached convergence. If no value is given, 10% of the trees present in the tree file will be removed. The value given in the control file is a number of trees, not a proportion or a number of generations.

```
burnin = 500
```

thinning [DEFAULT = 1]

This allows determines at which frequency you want to sample the trees post-burnin for the simulations. For example, a sampling frequency of 1 means that all trees will be used in the simulations, whereas a frequency of 2 means that every other tree will be used. This could be useful to reduce the amount of simulations if the number of trees in the file is important. The default is to sample all trees.

```
thinning = 1
```

seed [DEFAULT = -1]

This allows to enter a specific seed in order to be able to exactly replicate the simulations with the same results. Entering -1 will give a random seed, which is the default if the line is ommited from the control file.

```
seed = 12345
```

Original sequence file (optional)

You can provide the original sequence file in phylip format. It is important that the name of all sequences start with the species names as in the control file, immediately followed by a number that distinguishes the sequences from others from the same species. The number should start from 1 (e.g., chimp1, chimp2, ...). If you provide such a file, JML will calculate the minimum pairwise sequence distance between all pair of species and it will calculate the exact probability of observing them. These results are written in the file Probabilities.txt. If you provide a sequence file, JML will also ouput a file that will contain all pairwise distances for which the p-value is smaller than the significance level specified in the control file (default is $p<0.1$). This is useful to test for the overall fit of the model.

Running JML

Once the jml.ctl file is adequately completed and located in the same folder as the executable file and the species trees file, you are ready to run JML. JML will read the control file and then the tree file. By default, JML assumes that the species trees are in a file called 'species.trees' and that the control file is named 'jml.ctl'. If it can not find such a file, it will prompt you for the correct name of the file that contains the species trees. You can also give the name of the file from the command line using the '-t' flag. Same thing for the control file with the '-c' flag and the sequence file via the command line with the '-d' flag. For example, the command

```
$./jml -c control.txt -t mytrees.tre -d mysequences.phy
```

indicates that the control file is located in the file 'control.txt', that the species trees are located in the file 'mytrees.tre', and that 'mysequences.phy' is the original sequence file.

After reading the control file, JML will perform the simulations and calculate the posterior predictive distributions for the uncorrected minimum distance between sequences for all pair of species. Please look at the values output by the program in the buffer to make sure that all parameters are ok.

Example datasets are provided with the program in the folder /jml/example/. These should run correctly with your compiled program. Note that you might have to save all the files with the proper end-of-line signal for your operating system (Windows, Linux, Mac).

Ouput files

Distributions.txt

This file contains the posterior predictive distributions, which can be used to calculate the *p*-values of hybridization hypotheses. Each column represent one species comparison (species compared can be found in the first row). The simulated values are ordered from the smallest to the largest within a column, which makes it easy to calculate *p*-values. For example, if the observed uncorrected minimum distance between two species is 0.001, to obtain the probability of this observation you only have to count how many times a distance smaller or equal to this distance was obtained in the simulations and divide it by the total number of simulated values. For instance, if 27 values were smaller or equal than 0.001 and you performed 1000 simulations, the probability of observing a minimum distance of 0.001 or smaller in a scenario without hybridization is $27/1000 = 0.027$.

Probabilities.txt

This file is only output if a sequence file was provided. It contains the probability of observing the minimum distance between all pair of species according to a scenario without hybridization.

Results.txt

This file is only output if a sequence file was provided. It contains all the pairwise sequence distances that have a *p*-value smaller than the significance level specified in the control file (default is $p<0.1$). This is useful to find potential instances of hybridization in exploratoty approaches and to investigate whether the no hybridization model fit the data well (see *Interpretation of the results* section below). Note that if the file is empty, this means that no distances were found with a *p*-value below the threshold.

Files 'out.trees', 'rep.dat', and 'rep.phy'

These three files are constantly written to and read from during the simulations. You should not erase them while the program is running, but you can delete them once the simulations are over.

Interpretation of the results

Different approaches can be used for interpreting results from posterior predictive checking. An intuitive one is to interpret the p-value(s) directly. The p-values estimated by JML are posterior probabilities [2] and can be interpreted as the probability that the model will generate a minimum distance between sequences of two species smaller than that observed from the data, given the data. However appealing is this interpretation, it could lead to statistical issues when multiple tests are performed. Indeed, the need to correct for multiple statistical testing (Rice 1989) diminishes the likelihood of finding statistically significant results. This is especially problematic for the present application because the large variance in mutation rate for short sequences, combined with the difficulty to get long nucleotide sequence stretches that lack evidence of recombination in practice, result in power issues [1]. The problem is even more acute when the approach is used in an explorative way, that is if there are no a priori hypotheses of hybridization to test and if JML is only used to investigate the presence of hybridization in the dataset. In such cases, all pairwise species comparisons can be tested simultaneously and the statistical power will be greatly affected. To minimize power issues it could thus be important to specify hybridization hypotheses a priori without reference to the sequence data.

There is an alternative interpretation of posterior predictive checking, which is to see "how particular aspects of the data would be expected to appear in replications" [2]. For instance, we could evaluate the overall adequacy of a model by assessing if there are some aspects of the data that are not well predicted by the model. To do this, it would be of interest to report all observed distances that have a low probability of being observed, e.g. distances with $p < 0.1$ (this value is arbitrary and can be fixed by the user). This could indicate species comparisons where the model cannot adequately predict the observed minimum distances. If there were several of those instances, one could thus conclude that a strictly bifurcating species tree model is not adequate, probably because of the presence of hybridization. Note, however, that this is not the same as concluding that there has been hybridization between two given species. With such interpretations of posterior predictive distributions, the type I error is less of a concern because we use posterior predictive checking to evaluate the fit of the model rather than to test a specific hypothesis [2].

Regardless of the multiple comparison issues associated with posterior predictive checking, there are two points that should always be kept in mind when interpreting results from JML. First, posterior predictive checking is a test of the model and not of hybridization. If one rejects the model (bifurcating species tree without gene flow), this may well be because of the presence of hybridization, although it could also be due to other properties of the data such as undetected gene duplication, population substructure along the branches of the phylogeny, and parallel evolution. The second point to take into account is that a lack of evidence for hybridization with JML should not be interpreted as an absolute absence of hybridization in the dataset because (1) a lack of statistical significance can also be caused by a lack of data and that (2) not all hybridization events leave a detectable molecular signature [8, 1].

Non posterior predicting checking applications

Several users are tempted to use JML in the following way. First, they observe that one marker that has an incongruent evolutionary history compared to others (frequently, the mtDNA is incongruent relative to nuclear markers). They then want to test if the evolutionary history of the mtDNA marker is significantly different from that of the nuclear genes. To do this, they reconstruct a species tree with the nuclear genes and then test using JML if you can reject a scenario of incomplete lineage sorting for the mtDNA marker.

This type of use is completely different in phylosophy than the standard posterior predictive checking approach of JML. It assumes that the species tree (here reconstructed from nuclear genes) is the true species tree and that the population sizes and divergence times of the tree are accurate. If this is the case, then there is no problem in using JML this way. But if the species tree is not exact, then the results might be highly biased.

For this reason, I do not personnally recommend such an approach. I think the posterior predictive approach, which is an approach that checks for model adequacy, is more appropriate and conservative. It also does not assume that the reconstructed species tree is accurate. It just tests whether a bifurcating species tree with no migration is a good fit for the data. It is true that the results obtained might not be as significant, but “introgressed” sequenced should become more easily identified as more genes are included in the analysis.

Memory issues

JML requires a lot of memory to store the numerous distances matrix used to generate the posterior predictive distributions. Consequently, it is possible that the program runs out of memory. So if the program crashes unexpectedly while performing the simulations, it is quite likely that it is because the program has run out of memory. If this happens, try running the software on a computer that has more virtual memory or reduce the number of simulation performed (e.g., by sampling the MCMC chain less frequently in JML).

The source code of the program is also presently set up to allow a maximum of 50 species. If your dataset has more species, you will need to change the value of ’NSPECIES’ in the file MCMCcoal.c and recompile the program.

Frequently Asked Questions (FAQ)

Several pairwise species tests are reported in the output file "Probabilities.txt". Should I change anything to the input files if I only want to test a few pairwise species comparisons? I worry about the issue of multiple statistical testing.

Although JML reports statistical tests for all pairwise species comparison in the output files, you do not have to consider them all. The issue of multiple statistical testing is a philosophical one. If you choose a priori to test only some hybridization hypotheses, you should only consider these pairwise comparisons in the output files. The other comparisons, although computed, do not matter if they are not taken into account. Remember that the issue of simultaneous statistical testing is that if you make several statistical tests at a significance level of 5%, then you should expect that about 5% of the significant results might represent false positives. This is why the number of tests considered is important and should be decided a priori, i.e. without previously looking at the results.

For instance, imagine that you have one species for which a gene is strongly incongruent with the other genes. Then, a logical hypothesis to test would involve only this species and the species with which this species cluster in the "incongruent" gene phylogeny. So even if there are 100 species in your dataset (and thus N(N-1)/2 = 4950 species comparisons), you need only to consider one result in the output file. If more than one hypothesis has to be tested, then you should correct the significance results with an appropriate method (e.g., a sequential Bonferroni correction). JML outputs all the results at once to leave the decision to the user.

Citing JML

Please cite the following publication when using JML:

> Joly S. 2012. JML: testing hybridization from species trees. Molecular Ecology Ressources. 12:179–184.

As well as the original description of the method:

> S. Joly, P. A. McLenachan, and P. J. Lockhart. 2009. A statistical approach for distinguishing hybridization and incomplete lineage sorting. The American Naturalist. 174:e54-e70.

Do not forget to look at the JML website for software updates.

References

[1] Simon Joly, Patricia A. McLenachan, and Peter J. Lockhart. A statistical approach for distinguishing hybridization and incomplete lineage sorting. *The American Naturalist*, 174(2):e54–e70, 2009.

[2] Andrew Gelman, John B. Carlin, Hal S. Stern, and Donald B. Rubin. *Bayesian data analysis*. Texts in statistical science. Chapman & Hall, Boca Raton, FL, 2nd edition edition, 2009.

[3] Joseph Heled and Alexei J. Drummond. Bayesian inference of species trees from multilocus data. *Mol Biol Evol*, 27(3):570–580, March 2010.

[4] Alexei J. Drummond and Andrew Rambaut. BEAST: bayesian evolutionary analysis by sampling trees. *BMC Evolutionary Biology*, 7:214, 2007.

[5] Ziheng Yang. MCMCcoal: markov chain monte carlo coalescent program, December 2007.

[6] Andrew Rambaut and Nicholas C. Grassly. Seq-Gen: an application for the monte carlo simulation of DNA sequence evolution along phylogenetic trees. *Computer Applications in the Biosciences*, 13:235–238, 1997.

[7] Ziheng Yang. PAML 4: a program package for phylogenetic analysis by maximum likelihood. *Molecular Biology and Evolution*, 24:1586–1591, 2007.

[8] Simon Joly and Anne Bruneau. Incorporating allelic variation for reconstructing the evolutionary history of organisms from multiple genes: an example from rosa in north america. *Systematic Biology*, 55(4):623–636, 2006.