pr4design: Using probabilities of correctly resolving a split as an experimental design criteria Version 1.0

Edward Susko

Department of Mathematics and Statistics, Dalhousie University

Introduction

The main programs, `pr4design`, `pr4addbranch`, `pr4deltaxa` and `pr4list` implement the methods described in Susko and Roger (2012); please cite this reference when using the software. The main program, `pr4design` gives the probability of a split of interest in a newick format tree file being correctly resolved; `pr4list` is used to obtain the label of the split of interest. The programs `pr4addbranch`, `pr4deltaxa` allow manipulation of an initial tree to explore potential locations for adding or deleting taxa.

Installation

The main programs, `pr4design`, `pr4addbranch`, `pr4deltaxa` and `pr4list` are compiled from C and Fortran 77 source code. They utilize the Fortran 77 code developed by Alan Genz to implement the numerical integration algorithms described in Genz (2004).

In cases where there are a large number of taxa or amino acid data is being considered, it will usually be necessary to use Monte Carlo approximations of the expected information matrices utilized in calculations. The latter option requires that the program `seq-gen` (Rambaut and Grassly 1997) is available. This program can be downloaded from the aesthetically pleasing web site

`http://tree.bio.ed.ac.uk/software/seqgen/`

To install the `pr4design` routines on Mac or Unix systems

1. Download and unpack the software:

   ```
   $ tar zxf pr4design.tar.gz
   ```

   This will create a directory `pr4designv1.0` that contains the source code and test input files.

2. Change directories to `pr4designv1.0` and create the main program files with the `make` command:

   ```
   $ cd pr4designv1.0
   $ make
   ```

This should produce the program files `pr4design`, `pr4addbranch`, `pr4deltaxa` and `pr4list` which can be copied to a location in your PATH.

The source code has been compiled and tested using gcc and gfortran (versions 4.4.0) on an CentOS Linux distribution (release 5.5). While the program has not been tested on another platform, it should compile under any Linux distribution as well as Mac OS X, assuming a fortran compiler is installed.

Obtaining split/branch labels

The program `pr4design` obtains the probability of correctly resolving a split for an original tree or after adding taxa to and deleting taxa from that tree. In order to specify the split of interest, as well as possible splits/branches to add taxa to or from, it is necessary to specify which of the branches these are. The program `pr4list` can be used to accomplish this and can be run at the command line with the command

```
$ pr4list ntaxa treefile outtreefile
```

Here `ntaxa` is the number of taxa and `treefile` is the name of a file containing the original tree in Newick format.

A new tree with labels on internal branches will be printed to `outtreefile`. Newick format is used so that common tree drawing programs will bring up the tree with branch labels. For example with the program `njplot` (Perrière and Gouy 1996) available at

```
http://pbil.univ-lyon1.fr/software/njplot.html
```

the tree can be brought up with

```
$ njplot outtreefile
```

Clicking on the button labelled “Bootstrap” will indicate the labels of branches along them. Alternatively, the split labels can be obtained from the output to the screen which, for each split, lists the taxa on either side of the split.

As a running example we consider the seed plant data of the paper. To obtain the list of splits in the estimated tree of 11 taxa, stored in `seed_plant.tree`

```
$ pr4list 11 seed_plant.tree seed_plant_label.tree
Split 0: Psilotum from rest
Split 1: Amborella from rest
...
Split 18
Psilotum Amborella Nymphaea Magnolia Arabidops Oryza Zea
split from
Welwitsch Pinus Zarnia Cycas
```

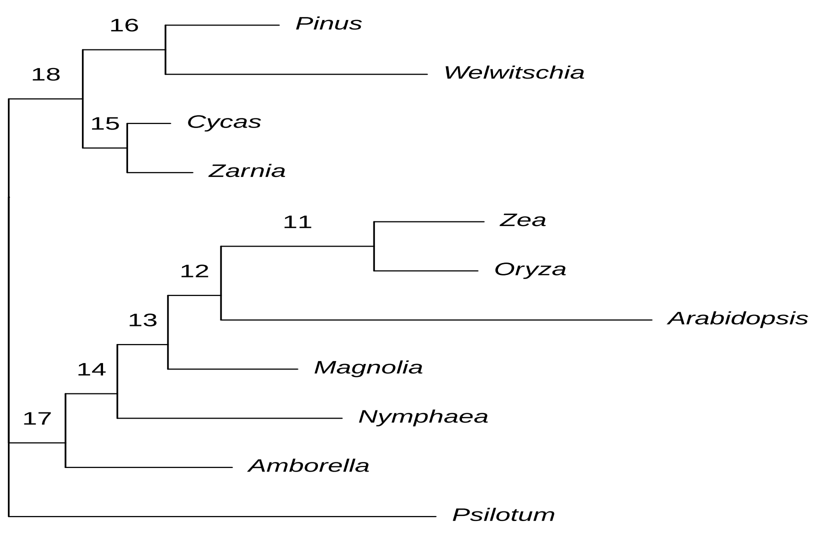

Figure 1: The seed plant tree with labels.

The command below, in conjunction with clicking on the button labelled “Bootstrap” produces a plotted tree similar to the one in Figure 1.

```
$ njplot seed_plant_label.tree &
```

Modifying the tree

The program `pr4deltaxa` can be used to delete taxa from the original tree and the program `pr4addbranch` can be used to add taxa. The labelling of the edges will change after running either of these programs. Thus `pr4list` will need to be run again to determine the labels of the splits in the new tree.

To delete taxa, use the command

```
$ pr4deltaxa number_of_taxa names < treefile
```

Here `names` is a space-separated list of names that are present in the Newick format tree file given by `treefile`. Output is to the screen and is a Newick tree file with the taxa removed. For instance, the command

```
$ pr4deltaxa 11 Arabidopsis Magnolia < seed_plant.tree > seed_plant_small.tree
```

deletes the taxa labeled `Arabidopsis` and `Magnolia` from the tree, with the resulting new tree stored in `seed_plant_small.tree`.

To add a branch, issue the command

```
$ pr4addbranch -n number_of_taxa -s split -a add_label -l length -x fraction
```

Here `split` is the label of the split that will be of interest in design, `add_label` is the label that a new branch is to be added to and `length` is it's length. The variable `fraction` indicates where along the edge for addition, the new edge will be added to. For values of `fraction` close to 0, the new branch is placed close to the end of the branch at the end that is furthest from the split of interest in design. For values of `fraction` close to 1, the new branch is placed close to the end of the branch at the end that is closest to the split of interest in design.

As an example,

```
$ pr4addbranch -n 11 -s 14 -a 17 -x 0.9 -l 0.1 < seed_plant.tree > seed_plant_large.tree
```

adds a branch of length 0.1 to the tree stored in `seed_plant.tree` 90% of the way along the 17th edge, away from edge 14. As another example,

```
$ pr4list 11 seed_plant.tree
Split 0: Psilotum from rest
...
$ pr4addbranch -n 11 -s 14 -a 0 -x 0.9 -l 0.1 < seed_plant.tree
```

adds a branch to the terminal edge leading to `Psilotum`.

Obtaining the probability of estimating the correct split

The program `pr4design` gives the probability of estimating the correct split and can be run at the command line with the command

```
$ pr4design controlfile
```

Input

All input to the routine is through a main control file, `controlfile`. The control file is similar in format to the control files used by the programs `baseml` and `codeml` in the PAML package (Yang 1997, 2007). For instance, the `model` variable specifies the substitution model and gives a subset of the models available in PAML, with the same numbering scheme. For the running seed plant example, and the original tree, with split 14 in Figure 1, as our split of interest, we use the control file `seed_plant.ctl`

```
* Example control file
splitofinterest = 14          * split of interest
middle = 0.002                * middle edge length

ntaxa = 11                    * number of taxa
treefile = seed_plant.tree    * Tree file (Newick format)

nchar = 4                     * nucleotide data
model = 7                     * GTR model
Qfile = seed_plant_GTRuQ      * file with rate matrix
nsite = 5544                  * number of sites for probability calculation
```

As with PAML control files, blank lines are allowed and all text following a '*' till the end of a line is treated as a comment. The word on the left of an equal sign gives a control variable and the word on the right gives the value of that variable. Spaces are required on both side of an equal sign. The order of variables is unimportant. The control variables are as follows. All variables not indicated as optional are required.

`splitofinterest`: An integer giving the split of interest for probability calculation. The probability approximation is the probability that this particular split of interest will be correctly estimated by maximum likelihood estimation. The labelling of splits can be obtained from `pr4list` as was described above.

`ntaxa`: An integer giving the number of taxa.

`treefile`: The name of a file containing the tree of interest with edge lengths. In the running example, the treefile was `seed_plant.tree` and contained the tree

```
(Psilotum:0.346,((Welwitschia:0.212,Pinus:0.092):0.067,
(Zarnia:0.053,Cycas:0.035):0.036):0.06,
(Amborella:0.135,(Nymphaea:0.182,(((Oryza:0.084,Zea:0.089):0.124,
Arabidopsis:0.349):0.043,Magnolia:0.105):0.041):0.042):0.046);
```

The tree should conform to the Newick standard. The programs in PHYLIP (Felsenstein, 1989, 2004), TREE-PUZZLE (Schmidt et al. 2002) and PAML (Yang 1997, 2007), which can be used to obtain ML estimates of edge-lengths for the models described here, will output trees in this format. A discussion of this standard as implemented in PHYLIP is given at

`http://evolution.genetics.washington.edu/phylip/newicktree.html`

and a more formal description is available at

`http://evolution.genetics.washington.edu/phylip/newick_doc.html`

Allowable features of the Newick standard that will likely create difficulties are:

1. Quoted labels.
2. Nested use of the characters '[' and/or ']' in comments. The characters '[' and ']' can only be used to delimit comments and cannot be used within comments.
3. Long leaf labels. A limit of 10 non-null characters is allowed for leaf names.
4. Underscores are not converted to blanks.

`nsite`: The number of sites for the probability approximation. For instance, if `nsite` is 1000, what is calculated is the probability of estimating the correct split when sequence length is 1000 character states (nucleotides or amino acids).

`nchar`: An optional integer indicating that the model was for nucleotide data (`nchar`=4) or amino acid data (`nchar`=20). The default value is 4.

`model`: An integer code for the substitution model. For nucleotide data (`nchar`=4), the models currently implemented are

`model`	Model
0	JC
2	F81
3	F84
4	HKY
7	GTR

and for amino acid data (`nchar`=20) the models currently implemented are

`model`	Model
0	Poisson
1	Proportional
2	Empirical
3	Empirical+F
8	REVaa

The documentation for the PAML package gives a good description of the models listed and can fit all of them.

The GTR and REVaa models refer to the most general time-reversible models in the nucleotide and amino acid case, respectively. The Poisson and Proportional models are the analogues of the JC and F81 models for amino acid data. The Poisson and Proportional models have substitution probabilities

$$P_{ij}(t) = \begin{cases} \pi_j + (1 - \pi_j)\exp[-\mu t] & \text{if } i = j \\ \pi_j - \pi_j \exp[-\mu t] & \text{otherwise} \end{cases}$$

where $\mu = [\sum \pi_i(1 - \pi_i)]^{-1}$ and π_j gives the stationary of the jth amino acid. In the Poisson model, the frequencies are all 1/20

When `model`=2 or 3 an empirical model is fit. The model is specified by the variable `aaRatefile`. When `model`=2, the stationary frequencies are the stationary frequencies of the specified empirical model.

When `model`=3, the stationary frequencies must be specified in a file `frfile`. In this case, the empirical model is used to specify the exchangeabilities. The exchangeability of amino acid i and j is defined as

$$S_{ij} = Q_{ij}/\pi_j$$

where, for the specified empirical model, Q_{ij} is the rate of substitution from i to j and π_j is the stationary frequency j. When `model`=3, the rate of substitution from i to j is

$$\tilde{Q}_{ij} = S_{ij}\tilde{\pi}_j$$

where $\tilde{\pi}_j$ is the frequency of j specified in `frfile`.

`aaRatefile`: Only required for empirical amino acid models (`model`=2 or 3 and `nchar`=20). The name of the empirical model to fit. The models currently implemented are

`dayhoff.dat`	Dayhoff or PAM	Dayhoff et al. (1978)
`jones.dat`	JTT	Jones et al. (1992)
`wag.dat`	WAG	Whelan and Goldman (2001)
`mtREV24.dat`	mtREV	Adachi and Hasegawa (1996)
`lg.dat`	LG	Le and Gascuel (2008)

The naming scheme was chosen to be consistent with PAML. However, `aaRatefile` is not actually the name of file, rather it identifies a model.

`frfile`: The name of a file containing the model's stationary frequencies nucleotides or amino acids, each separated by white space. For nucleotides, the ordering is alphabetical: A, C, G and T, which differs from the T, C, A and G ordering of PAML. Amino acids are ordered alphabetically:

```
alanine, arginine, asparagine, aspartic, cysteine,
glutamine, glutamic, glycine, histidine, isoleucine,
leucine, lysine, methionine, phenylalanine, proline,
serine, threonine, tryptophan, tyrosine, valine
```

which is the same ordering used by most phylogenetic packages including PAML, PHYLIP and TREE-PUZZLE.

`Qfile`: Only required for the general time reversible model, GTR or REVaa (`model`=7, `nchar`=4 or `model`=8, `nchar`=20). The name of a file containing the entries of the rate matrix separated by blanks. The ordering of nucleotides or amino acids should match the ordering in `frfile`. In the example, the entry in the control file is

```
Qfile = seed_plant_GTRuQ              * file with rate matrix
```

and the file `GTRuQ` contains the entries

```
-1.091117    0.585134    0.351278    0.154705
 0.811488   -1.208900    0.255487    0.141925
 0.313818    0.164577   -0.921570    0.443175
 0.170554    0.112821    0.546897   -0.830272
```

`kappa` or `ttratio`: One of these is required for the F84 and HKY models (`model`=3 or 4 and `nchar`=4). A real number giving the κ parameter for the model. The F84 model has a rate matrix proportional to

$$\begin{bmatrix} \cdot & \pi_C & (1+\kappa/\pi_R)\pi_G & \pi_T \\ \pi_A & \cdot & \pi_G & (1+\kappa/\pi_Y)\pi_T \\ (1+\kappa/\pi_R)\pi_A & \pi_C & \cdot & \pi_T \\ \pi_A & (1+\kappa/\pi_Y)\pi_C & \pi_G & \cdot \end{bmatrix}$$

where $\pi_R = \pi_A + \pi_G$ and $\pi_Y = \pi_C + \pi_T$. The HKY model has a rate matrix proportional to

$$\begin{bmatrix} \cdot & \pi_C & \kappa\pi_G & \pi_T \\ \pi_A & \cdot & \pi_G & \kappa\pi_T \\ \kappa\pi_A & \pi_C & \cdot & \pi_T \\ \pi_A & \kappa\pi_C & \pi_G & \cdot \end{bmatrix}$$

The transition-transversion ratio (`ttratio`) is related to the κ parameter in the F84 model through

$$R = \kappa \times \frac{\pi_A\pi_G/\pi_R + \pi_C\pi_T/\pi_Y}{\pi_R\pi_Y} + \frac{\pi_A\pi_G + \pi_C\pi_T}{\pi_R\pi_Y}$$

where $\pi_R = \pi_A + \pi_G$ and $\pi_Y = \pi_C + \pi_T$. For the HKY model the relationship is

$$R = \kappa \times \frac{\pi_A\pi_G + \pi_C\pi_T}{\pi_R\pi_Y}$$

`alpha`: Only required if a gamma rates-across-sites model (Yang 1994) is desired. A real value giving the shape parameter of the gamma distribution.

`ncatG`: Only used if a discrete gamma rates-across-sites model was fit. Optionally, an integer giving the number of categories to use in the discrete approximation. The default is 4. The discrete gamma approximation used is the same as the default of PAML 4.2; the representative rate is the conditional mean for the class.

`nrate`: Optionally used to specify the number of rate classes in an arbitrary discrete rates-across-sites model. The variable is not used if `alpha` is set. If `alpha` is not set the default is to fit an equal rates model.

`ratefile`: Only used if `alpha` is not set and `nrate`>1. The name of a file with rates and weights for the arbitrary discrete rates-across-sites model. Each line should give a rate followed by a corresponding weight. For example, the file

```
0.00026 0.25
0.02369 0.25
0.33291 0.25
3.64313 0.25
```

indicates a rates-across-sites model with four rates where, for instance, there is a 25% chance that a site will have a rate multiplier 3.64313. This consequently allows for implementations of the discrete gamma rates-across-sites models that give different discrete rates or weights than PAML to the same shape parameter.

`tinfo`: Optionally one of 0, 1 or 2. The default is 0.

The probability calculations use expected information matrices that can require substantial calculation when either the number of taxa is large or amino acid data is considered. These matrices can be approximated through simulation. The value `tinfo`=0 indicates that calculation should be exact and the value `tinfo`=1 indicates that simulation should be used for calculation. The value `tinfo`=1 assumes that the program `seq-gen` is available. Alternatively, if it is more convenient to simulate directly, simulated data can be read through the sequence file, `seqfile`. In this case `tinfo` should be set to 2.

`seqfile`: Only required if `tinfo=2`. The name of a file containing the sequence data that will be used for information matrix calculation when `tinfo=2`. For accurate approximation, this should be simulated with the same model that is being used in probability calculations.

The file should conform to PHYLIP standards for input with 10 character long names padded by blanks. The names should match the names used in the input treefile. Input can be either interleaved or sequential with one caveat: The lines 2 through $m+2$, where m is the number of taxa, must contain the name of taxa followed by sequence data. For instance the start of a sequence file might be

```
6 3414
Homsa      ANLLLLIVPI LI...
Phovi      INIISLIIPI LL...
...
```

but not

```
6 3414
Homsa
ANLLLLIVPI LI...
Phovi
INIISLIIPI LL...
...
```

which would be allowed under the sequential format by PHYLIP.

`nsim`: Optional and only used if `tinfo=1`. The number of sites to simulate when approximating the information matrix. The default is 100,000.

`iseed`: Only used if `tinfo=1`. An integer that will set the seed. Changing this will change the data generated for approximating the information matrix and can be useful in evaluating the sensitivity of the calculations to this random generation. If probabilities vary substantially, `nsim` should be increased.

Output

The output (to the screen or stdout) is the probability that the split is estimated correctly. For instance, with the running example, the control file set `splitofinterest` to 14 based on the labelling in Figure 1. The probability of correctly estimating this split is

```
$ pr4design seed_plant.ctl
6.5712739653234298e-01
```

A tree with two of the taxa deleted was created with

```
$ pr4deltaxa 11 Arabidopsis Magnolia < seed_plant.tree > seed_plant_small.tree
```

The labelling of the splits in the new tree might be different than for the original tree and needs to be checked.

```
$ pr4list 9 seed_plant_small.tree seed_plant_small_label.tree
...
Split 12
Psilotum Amborella Zarnia Cycas Welwitsch Pinus
split from
Nymphaea Oryza Zea
```

The original split 14 now has the label 12. To investigate effects of reducing the number of taxa but increasing the number of sites from 5544 to 7000, we copy the control file to `seed_plant_small.ctl` and edit it.

```
splitofinterest = 12            * split of interest
middle = 0.002                  * middle edge length
ntaxa = 9                       * number of taxa
treefile = seed_plant_small.tree * Tree file (Newick format)

nchar = 4                       * nucleotide data
model = 7                       * GTR model
Qfile = seed_plant_GTRuQ        * file with rate matrix
nsite = 7000                    * number of sites for probability calculation
```

Running `pr4design` again gives

```
$ pr4design seed_plant_small.tree
5.6007789621417259e-01
```

Change Log

Changes since March 1, 2012:

- September 5, 2014: Control files with carriage returns (which are automatically added when created on a Windows OS) are now correctly interpreted.

References

Genz A. (2004). Numerical computation of rectangular bivariate and trivariate normal and t probabilities. Statistics and Computing. 14:251–264.

Perrière, G. and Gouy, M. (1996) WWW-Query: An on-line retrieval system for biological sequence banks. Biochimie, 78, 364-369.

Rambaut, A. and Grassly, N. C. (1997). Seq-Gen: An application for the Monte Carlo simulation of DNA sequence evolution along phylogenetic trees. Comput. Appl. Biosci. 13: 235-238.

Susko, E. and Roger, A.J. (2012). The Probability of Correctly Resolving a Split as an Experimental Design Criterion in Phylogenetics. Syst. Biol. 61:811–821.

Yang, Z. (2007). PAML 4: a program for phylogenetic analysis by maximum likelihood. Mol. Biol. Evol. 24:1586–1591.

Yang, Z. (1997). PAML: a program for phylogenetic analysis by maximum likelihood. Comput. Appl. Biosci. 13:555–556.