


CAFE: Computational Analysis of gene Family Evolution

Documentation



Mar 14, 2017

Version 4.0

Contents

1	About CAFE	3
1.1	Citing CAFE	3
2	Installation	4
2.1	Downloading	4
2.2	Compiling	4
3	Running CAFE	6
3.1	Commands	6
3.2	Output	20
4	Troubleshooting	22
4.1	Known limitations	22
4.2	Looking for help	23
4.3	License	23

1 About CAFE

CAFE is a software that provides a statistical foundation for evolutionary inferences about changes in gene family size. The program employs a birth and death process to model gene gain and loss across a user-specified phylogenetic tree, thus accounting for the species phylogenetic history. The distribution of family sizes generated under this model can provide a basis for assessing the significance of the observed family size differences among taxa.

Throughout the years, many people contributed to the CAFE project (see below). Version 4.0 has received additional contributions from: Ben Fulton, Fábio K. Mendes, Matthew W. Hahn and Robert Henschel.

1.1 Citing CAFE

The citation for CAFE v3.1 and v4.0 is:

HAN, M. V., THOMAS, G. W. C., LUGO-MARTINEZ, J., AND HAHN, M. W. Estimating gene gain and loss rates in the presence of error in genome assembly and annotation using CAFE 3. *Molecular Biology and Evolution* 30, 8 (2013)

The citation for CAFE v2.0 is:

HAHN, M. W., DEMUTH, J. P., AND HAN, S.-G. Accelerated rate of gene gain and loss in primates. *Genetics* 177 (2007)

Original development of the statistical framework and algorithms implemented in CAFE are published in:

HAHN, M. W., DEBIE, T., STAJICH, J. E., NGUYEN, C., AND CRISTIANINI, N. Estimating the tempo and mode of gene family from comparative genomic data. *Genome Research* 15, 8 (2005)

2 Installation

2.1 Downloading

CAFE can be downloaded in two different ways:

1. Go to <https://hahnlab.github.io/CAFE/download.html> to find a download link to the tarball. Then put the tarball in a folder of your choice, and from that folder extract the files with command (replacing “some_version” with the version you downloaded):

```
$ tar zxvf CAFE-some_version.tar.gz
```

2. If you have **git** installed, you can clone CAFE’s repository by moving to a directory of your choice and entering the command:

```
$ git clone http://www.github.com/hahnlab/CAFE.git
```

2.2 Compiling

CAFE v4.0 comes with a precompiled executable file for Linux machines. You can find the executable file, **cafe**, in `/path/to/CAFE/release/`. If you prefer to compile from source yourself, move to the CAFE folder, and run:

```
$ make
```

The **cafe** executable will be put inside the **release** folder. If the GNU readline library is installed on your system and you wish to use it to provide a command history for CAFE, you can instead compile with command:

```
$ make USE_READLINE=1
```

Then just copy the binary file to somewhere on your path, such as `/usr/local/bin`. Alternatively, add `/path/to/CAFE/release/` to your `$PATH` variable (in your `.bashrc` or `.bash_profile`).

3 Running CAFE

In order to give CAFE commands, you must do it interactively through its shell, or by providing CAFE with a shell script listing CAFE commands. To run CAFE interactively, type `cafe` at your shell prompt. If all is well, the prompt should change to `#`. At this point you may now begin inputting commands. To **exit** the shell, type `exit`.

If you need to run multiple analyses using similar inputs, you can provide CAFE with a shell script. These scripts should be saved as text files with UNIX line endings. Scripts may be then executed from your OS or CAFE's shell. Here is an example:

```
#!/cafe
#version
#date
load -i data/example2.tab -t 10 -l logfile.txt -p 0.05
tree (((chimp:6,human:6):81,(mouse:17,rat:17):70):6,dog:93)
lambda -s -t (((1,1)1,(2,2)2)2,2)
report resultfile
```

In this example, the first line indicates the location of the CAFE shell program. Subsequently, lines beginning with “`#`” are regarded as comments. Thus, the example above only executes lines 4, 5, 6 and 7. Remember that to run a script you must make the file executable from your OS shell prompt (in the UNIX shell, this is done with `chmod a+x filename`). CAFE will automatically exit after the last command in the script is completed, so it is not necessary to specify the `exit` command.

3.1 Commands

Below you will find a table summarizing the commands CAFE recognizes:

Command	Short description
<code>cafererror.py</code>	Estimate error in data file
<code>date</code>	Display date/time
<code>errormodel</code>	Apply error correction to data
<code>esterror</code>	Estimate error matrix from multiple datasets
<code>exit</code>	Exit CAFE shell
<code>genfamily</code>	Generate simulated data
<code>lambda</code>	Find/specify birth-death parameter
<code>lambdamu</code>	Find/specify separate birth and death parameters
<code>lhstest</code>	Compare likelihoods of lambda models
<code>load</code>	Data file & run parameters
<code>log</code>	Log output
<code>noerrormodel</code>	Remove applied error models
<code>report</code>	Report values
<code>rootdist</code>	Specify root family size distribution for simulation
<code>simerror</code>	Simulate/add error to data based on error model
<code>source</code>	Run shell script
<code>tree</code>	Phylogenetic tree
<code>version</code>	Version info

Detailed descriptions of the commands can be found below:

```
$ python cafererror.py [-i shell script filename] [-e initial error value]
[-d output directory name] [-l log filename] [-o output filename] [-s value]
```

`cafererror.py` is a Python script included with the CAFE v4.0 software package that uses the `errormodel` command iteratively to estimate error in an input data set with no prior knowledge of the error distribution. `cafererror.py` uses the likelihood scores of runs with varying error models to perform a precise grid search of the likelihood surface. The program first estimates average global error across all species in the input phylogeny and then may continue to individual species estimations depending on `-s`.

Note that:

1. Python 2.6 or newer must be installed on your machine. You can find it on <https://www.python.org>;
2. `cafererror.py` must run in the directory in which CAFE is located, as it uses that path to run CAFE.

-i *shell script filename*: The main input of `cafererror.py` is a CAFE shell script, as shown above. `cafererror.py` will extract the following information needed from the shell script and use it to run CAFE many times to estimate error: the input gene family file from the `load` command, the `tree` command, and the `lambda` command. `cafererror.py` will not overwrite the input script, but will instead write its own.

-e *initial error value*: This is the value with which `cafererror` will begin the grid search. This should be a floating point value between 0 and 1. Default: 0.4.

-d *output directory name*: `cafererror.py` runs CAFE many times, and therefore creates and stores many error model and CAFE log files. All CAFE log files, error model files, and `cafererror.py` output files will be stored in a directory specified with this option. If the directory has not been created, `cafererror.py` will create it automatically. Default: `cafererror_tmp_dir_x`, where *x* is an integer one higher than the previous default directory.

-l *log filename*: `cafererror.py` keeps track of the error estimates and scores in its own log file. This is also where you will find the final error estimates. The user may specify the name of the file with this option. Default: `cafererrorLog.txt`.

-o *output filename*: The user may specify a name for the output file with this option. The error estimation algorithms create a curve for visualization if plotted, and this output file contains two tab-delimited columns consisting of error model and the corresponding score while using that error model. Simply copy and paste these data points into your favorite graphing software to see how `cafererror.py` estimated the error. Default: `cafererror_default_output.txt`.

Note: this option outputs data points for the global error estimation.

date

Display current date and time.

errormodel [-model *filename*] [-sp *species name* | -all]

The **errormodel** command allows the user to specify an error distribution. CAFE will correct for this error before calculating ancestral family sizes and estimating λ values. The **errormodel** function is also used by **cafeerror.py** to estimate error in the input data set.

-model *error model file*: This option allows the user to specify the error file to use in order to correct the input data for errors. The error model file format should be as follows:

```
maxcnt: 68
cntdiff -1 0 1
0 0.0 0.8 0.2
1 0.2 0.6 0.2
2 0.2 0.6 0.2
...
68 0.2 0.6 0.2
```

In this file, **maxcnt** is the largest family size observed in the dataset. Error classes (for all following rows) are defined with **cntdiff** and act as labels for error distributions for each gene family size. Error classes must be space-delimited positive or negative integers (and 0). The error class with label 0 means that this corresponds to no change in gene family size due to error. After the first two lines, each possible *family size* in the dataset (size 0 to **maxcnt**) should have an error distribution defined. Any omitted family size follows the distribution for the previous row. The error distribution for each count should be space delimited probabilities whose columns correspond to the error classes defined in line two. Default: No error model is applied.

Note that:

1. You should not specify any negative error correction for family size of 0 as this cannot occur (i.e., there can't be negative gene family sizes);
2. The rows of the error model file must sum to 1;
3. If any gene counts are missing from the error model file, CAFE will assume the same error distribution from the previous line. This can also be used as a shortcut if you know that all of the gene counts are specified with the same error distribution: simply enter the first four lines (`maxcnt`, `cntdiff`, `family size=0,1`) into the error model file and CAFE will use the distribution for *family size=1* as the distribution for all gene family sizes.

-sp: This option is required to specify the species to which the error model will be applied. Species names must be identical to those in the data file and the input tree. The user may specify any combination of species with the same or different error model files with separate `errormodel` commands, or the user may specify all species with the same error model file in one `errormodel` command using `-all` as the species option here.

```
esterror [-dataerror filename -datatrue filename] | [-dataerror filename1  
filename2] [-diff matrix size] [-symm] [-o output filename]
```

`esterror` estimates the error matrix from two different gene family datasets. If the user gives two `-dataerror` options, the error is estimated assuming two error prone measures. If the user gives one `-dataerror` option and one `-datatrue` option, the error is estimated assuming `-datatrue` is the true measure and `-dataerror` is the error prone measure.

-diff *matrix size*: *matrix size* is the number of rows away from the main diagonal to which error matrix parameters are constrained. Default: 2.

-symm: Constrains the error matrix to be symmetrical between the upper triangle and the lower triangle. Default is an asymmetrical model.

-o *output filename*: Specifies the output file that the estimated error model will be written to. The user can then use *output filename* with the command **errormodel** to set the estimated model to the appropriate species.

exit

Exit the CAFE shell (**quit** will perform the same action).

genfamily [*directory name/filename prefix*] [-t # of simulated datasets]

The **genfamily** command generates simulated data based on the properties of observed data. These simulated data can be used for many purposes, including generating null distributions of likelihood ratios to assess the significance of multi-parameter models (see **lhtest**). CAFE uses the estimated root sizes from the observed data, the most recently specified λ and μ (either by search or user input), and the tree specified in the **tree** command to generate new data sets. Each simulated data file will contain the same number of families and distribution of root sizes as the observed data (unless otherwise specified with the **rootdist** command). To specify the data file and the value of λ , the **load** and **lambda** commands must precede **genfamily**.

directory name/filename prefix: Designates the directory where CAFE will write the simulated data sets. This should be specified relative to the working directory, and must be created prior to running CAFE (i.e., CAFE will not create the directory). Each simulated data set will have the name *fileprefix_#.tab*.

-t # of simulated datasets: Specifies the number of simulated data set for CAFE to generate.

For example, the command **genfamily rndtree/rnd -t 100** will generate the simulated datasets: *rnd_1.tb*, *rnd_2.tb*, ..., *rnd_100.tb*, and write them in the *rndtree* directory. These datasets will have the same format as typical CAFE input.

lambda [-l values | -s | -r *start:step:end*] [-t *lambda structure*]

-l *lambda values*: This option allows the user to specify the value(s) of λ . If more than one λ is specified, then the **-t** option must also be used. λ values are specified in the order 1 2 3... which correspond to the integers specified in the **-t** option. λ values should be separated by spaces.

Note that:

1. The product of λ and the depth of the tree structure should not exceed one (i.e., $\lambda \times t < 1$ must be true; where t is the time from tips to the root). See the [Known limitations](#) section for details as to how to diagnose this problem.
2. CAFE will use the last λ value(s) estimated (or user-specified) to compute ancestral gene family sizes and to run Monte Carlo simulations.

-s: CAFE will search using an optimization algorithm to find the value(s) of λ that maximize the log likelihood of the data for all families. CAFE starts with an intermediate value and then searches iteratively for the best value for λ (or set of λ values if used in conjunction with the **-t** option). Subsequent analyses will automatically use the results from the lambda search.

-r *start:step:end*: Returns the likelihood scores for λ values in the user specified range (**start:step:end**). For example, to see the score distribution with a lambda between 0.003 and 0.005, the range would be 0.003:0.001:0.005. In case of more than one lambda, ranges are separated by a space.

-t *lambda structure*: To investigate whether different parts of the tree are evolving at different rates, the user may specify which branches of the tree will take the same of different λ values. Input the same NEWICK tree structure as in the **tree** command, but exclude branch lengths and substitute integer values from 1 up to n or taxon names (where n = the total number of branches on the tree; matching integer values indicate that these branches will take the same value of λ). Default: all branches have the same value for λ . Here is an example of a lambda structure:

$(((1,1)1, (2,2)2)2,2)$

This λ structure corresponds to the tree example above, and specifies $\lambda = 1$ for the Human, Chimp, and Ape branches, and a $\lambda = 2$ for the remaining branches.

Note: CAFE will not always converge to a single optimum with models that contain many parameters. See the [Known limitations](#) section below for details on assessing this problem.

`lambdamu [-l lambda list -m mu list | -s] [-t lambda structure] [-eqbg]`

`-l lambda values`: Identical to `-l` for the `lambda` command, but now must be used in conjunction with `-m`.

`-m mu values`: This option allows the user to specify the value(s) of μ , the death rate.

Note that:

1. The product of λ or μ and the depth of the tree should not exceed one (i.e., $\lambda \times t < 1$ and $\mu \times t < 1$ must be true; where t is the time from tips to the root). See the [Known limitations](#) section for details as to how to diagnose this problem.
2. CAFE will use the last λ/μ value(s) estimated (or user-specified) to compute ancestral gene family sizes and to run Monte Carlo simulations.

`-s`: Identical to the `-s` option from the `lambda` command, CAFE will search for separate birth (λ) and death (μ) rates if the `lambdamu` command is specified.

CAFE starts with an intermediate value and then searches iteratively for the best value for λ (or set of λ values if used in conjunction with the `-t` option). Subsequent analyses will automatically use the results from the `lambda` search.

-t *lambda structure*: Identical to the **-t** option from the **lambda** command, but used in conjunction with the **lambdamu** command the tree structure can specify branches on which to estimate separate birth (λ) and death (μ) rates. In other words, branches with the same numerical identifier will share λ and μ parameters. See the **-t** section in the **lambda** command for a sample λ structure. Default: all branches have the same value for λ and μ .

-eqbg: Only used when the λ structure is specified with **-t**. This option allows the user to constrain the background rate (those branches with numerical identifier “1”) to have $\lambda = \mu$, while other branches are allowed separate estimates of λ and μ .

lhstest [-d *directory name*] [-l *lambda seed value*] [-t *lambda structure*]
[-o *output filename*] [# of replicates]

The **lhstest** command computes two likelihood scores for each simulated data set: one based on a model with a single, global λ and one based on a model with multiple λ values. Using simulated data with one λ parameter from **genfamily**, the results of **lhstest** can be used to generate a distribution of likelihood ratios [i.e., $LR = 2 \times (\text{score of global } \lambda \text{ model} - \text{score of multi-}\lambda \text{ model})$] under the null hypothesis.

This distribution can then be used to assess the significance of models with more than one λ parameter in observed data, by comparing the likelihood ratio to the distribution of ratios generated by analysis of simulated data sets. A multi-parameter model is significantly better than a single-parameter model if the observed LR is greater than 95% (at $\alpha = 0.05$) of the distribution of simulated LRs.

-d *directory name*: Specifies the directory where CAFE can find the simulated datasets generated by the **genfamily** command.

-l *lambda seed value*: For each simulated data set, CAFE will begin the λ search algorithm at the value specified here. Since CAFE reestimates λ s for each simulated data set, specifying a seed value close to the actual λ used in the simulations may save considerable time.

-t *lambda structure*: Specify the λ structure as in the `lambda` command above. This command specifies the multi- λ model to be estimated.

-o *output filename*: The file where CAFE will write the output of `lhtest`. The file contains columns for each of the following values: likelihood score for global λ — estimated global λ — likelihood score for multi- λ model — estimated λ 1 — estimated λ 2 — ... — estimated λ n.

`load -i filename [-t # of CPU threads] [-l filename] [-p α] [-r # of random samples]`

-i *filename*: Enter the path to the file containing gene family data. The data file format must be tab-delimited with UNIX line endings. Family description may contain spaces (but not tabs). The first line must contain labels in the order: *Description*, *ID*, and then *tab-delimited taxon names* (NOTE: If you do not have a Description or ID, CAFE still requires two tabs at the beginning of each line). The taxon names must be spelled exactly as they are in the provided phylogenetic tree. Each subsequent line then corresponds to a single gene family. If the data file contains taxa that do not appear in the tree structure, they are not considered in the analysis.

Here is an example of an input data file:

Description	ID	Chimp	Human	Mouse	Rat	Dog
EF 1 ALPHA	ENSF000000000004	5	8	6	12	40
HLA CLASS II	ENSF000000000007	4	4	3	3	3
HLA CLASS I	ENSF000000000014	5	3	5	6	3
RAG 1	ENSF000000000015	1	1	1	1	1
IG HEAVY CHAIN	ENSF000000000020	32	42	51	60	18
ACTIN	ENSF000000000027	27	30	22	28	25
OPSIN	ENSF000000000029	2	2	2	2	2
HEAVY CHAIN	ENSF000000000030	25	25	23	24	18

If the file is loaded correctly, CAFE will output summary information about the current data file to the log file.

-t # of CPU threads: The maximum number (integer) of CPU threads to be used. Default: 8.

-l filename: Enter the path to the file where CAFE will write the main **output**. This file will contain a summary of input parameters as well as details of λ searches, including likelihood scores and maximum likelihood values of λ . If the file does not exist, CAFE will create it for you; if the file already exists, CAFE will append the results to the previous file. Default: output to screen (no log file created).

-p α : For each family in the data file, CAFE computes a probability (p-value) of observing the data given the average rate of gain and loss of genes. All else being equal, families with more variance in size are expected to have lower p-values. The significance level (α , a float) allows the user to specify the cutoff for subsequent analyses. Families with p-values larger than the designated significance level will not be included in the identification of the most unlikely branch. Default: 0.01.

-r # of random samples: To determine the probability of a gene family with the observed sizes among taxa, CAFE uses a Monte Carlo re-sampling procedure. This option specifies the number of samples CAFE should use to calculate p-values. The tradeoff is between precision and computation time; in most cases 1000 samples should provide reasonable balance. Default: 1000.

-filter: The birth-death model of CAFE assumes at least one gene in the root of the species tree. This assumption may not be valid for families that were created after the most recent common ancestor of all species. The **filter** option filters out the families that are inferred (by parsimony) to have no genes in the root node of the species tree.

log filename

If no *filename* is given, this command displays the current log file. If *filename* is specified, CAFE will create (or overwrite) the log file. Default: **stdout** (output to screen only).

The log file may also be specified in the load command using the `-l` option.

`noerrormodel [-sp species name | -all]`

`-sp`: This option is required to specify the species to which the error model should not be applied (i.e., no error model). The species names should be identical to those in the data file and input tree. To remove error models from all species, use `-all`.

`report filename`

The `report` command outputs results. Although all analyses must be specified by their own commands, `report` directs the output of CAFE to *filename* (there is no need to add an extension to *filename*).

Here is a description of the output file (a tab-delimited summary of the results):

- *Tree*: The current tree;
- *$\lambda(s)$ and likelihood*: The current λ values set by the `lambda` command; it can be either specified by the user (`-t`) or obtained by searching for the maximum likelihood value (`-s`). The likelihood of the data given the current λ value;
- *Average expansion*: Mean number of genes gained or lost per family, where “minus” expansion is a net contraction;
- *Expansions and contractions*: Total count of families that experienced expansions, contractions, or no change along each branch of the species tree;
- *List of family and description*;
- *List of overall p-value for each family*: The p-values are based on a Monte-Carlo re-sampling procedure. To determine the probability of a gene family with the observed sizes among taxa, CAFE will generate the expected distribution of family sizes under the stochastic birth-death model for the tree specified in the `load` command with the current λ value.

Running the simulations uses the most machine resources and thus is the most time intensive step in CAFE. For each family in the data file, CAFE computes a probability (p-value) of observing the data given the average rate of gain and loss of genes. All else being equal, families with more variance in size are expected to have lower p-values.

- *List of branch-specific p-values for the significant families:* The branch-specific p-values are obtained by the Viterbi method with the randomly generated likelihood distribution. This method calculates exact p-values for transitions between the parent and child family sizes for all branches of the phylogenetic tree. A low p-value indicates a rapidly evolving branch. This information is reported only for the families with an overall p-value less than the p-value cutoff set with the `load` command.
- *List of ancestral states for each family:* Reports the maximum likelihood values of the ancestral number of genes at all inner nodes of all gene families.

`rootdist [-i filename]`

Used before the `genfamily` command, this command allows the user to set the root family size distribution from which the simulations are run to generate artificial gene family sizes.

`-i filename`: The `rootdist` input file is a text file in the format shown below. The total sum of the frequencies should add up to the number of families the user is aiming to simulate.

The input file of the `rootdist` command should look like this:

Var	Freq	Max:	30
1	10697		
2	563		
...			
30	1		

Var Freq Max is the maximum root family size desired. Following that line, the root family size is defined as the first number of each row and the number of families to have that size is the second number in the row. Specifically, the above example would specify root distributions in the following way: 10697 families would have a root size of 1, 563 families would have a root size of 2, and so on.

`simerror [-p gain_diff_1/simerror] [-rep # of replicates]`

Simulates and adds error to gene family sizes according to the error model specified. The error is added to the data provided in the gene family input file; these data can either be simulated themselves or come from an external dataset. For this single input dataset, each replicate gets error randomly added to it. The command `load` must have been used before to load the data. The command `errormodel` must have been used before running this command to specify the error to simulate.

`-pre`: The prefix for the family size files generated by the error simulation.

`-rep # of replicates`: The number of replicates to be simulated. Default: 10.

`source filename`

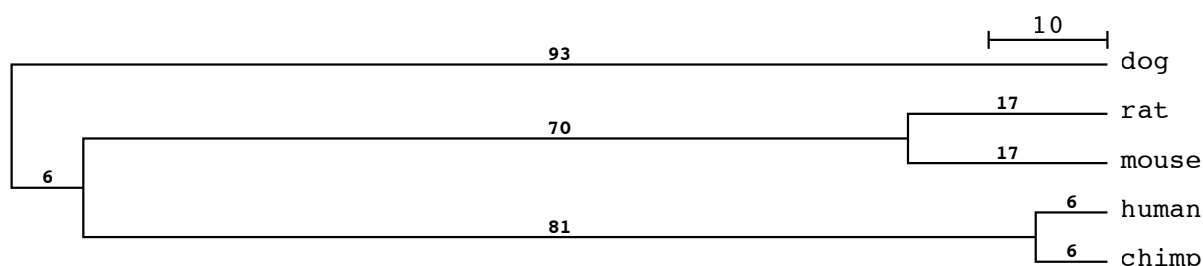
Load shell script file.

`tree NEWICK-formatted tree`

A NEWICK-formatted tree containing branch lengths and taxon names as they are specified in the input file. Branch lengths should be integer units and the tree should be ultrametric (all paths from root to tip should have the same length).

Please note that there should be **no spaces** in the tree string, nor semicolons at the end of the line.

Here is an example of an ultrametric tree followed by its NEWICK notation:



```
((chimp:6,human:6):81,(mouse:17,rat:17):70):6,dog:93)
```

version

Display the CAFE version number.

3.2 Output

CAFE's main output file will be written to a `.cafe` text file, the name of which will have been specified by the `report` command. An output file from running the `lambda` command on a dataset of mammalian gene families, for example, produces a file like the one below:

```
Tree:(((cat:68.7105,horse:68.7105):4.56678,cow:73.2773) (...) :96.4356)
(((cat:68.7105,horse:68.7105):4.56678,cow:73.2773) (...))
Lambda: 0.00265952
Lambda tree: (((((1,1)1,1)1,((((1,1)1,1)1,1)1,(1,1)1)1,1)1,1)1)1)
# IDs of nodes:(((cat<0>,horse<2>><1>,cow<4>) (...) (rat<20>,mouse<22>><21>><19>
# Output format for: (...) = (node ID, node ID): (0,2) (1,4) (...)
(...)
'ID' 'Newick' 'Family-wide P-value' 'Viterbi P-values'
8 (((cat_59:68.7105 (...) mouse_61:36.3024)_62:96.4356)_62 0.438 ((-,-),(-,-) (...))
10 (((cat_57:68.7105 (...) mouse_52:36.3024)_53:96.4356)_55 0.916 ((-,-),(-,-) (...))
11 (((cat_37:68.7105 (...) mouse_79:36.3024)_69:96.4356)_52 0 ((0.0699637,0.487686),(0.758569,0.202491)
(...))
```

Some of this output is self-explanatory, but let us point out what is important:

- On the second line, you will find the estimated value of λ for the whole tree, which for this run of CAFE was 0.00265952.

- The line that starts with ‘# IDs of nodes’ gives us the number that will represent each species and internal node. So for this run of CAFE, for example, ‘0’ represents cat, ‘2’ represents horse, and so on.
- The line that starts with ‘# Output format for:’ shows the pairs of species or internal branches for which results will be presented later. Whenever you see pairs of values enclosed in parentheses, one after the other, they shall follow the order shown on this line. In the example above, the first pair of values enclosed in parenthesis you see will refer to cat and horse (‘(0,2)’), the second pair of values will refer to the internal branch subtending (cat,horse) and then cow (‘(1,4)’), and so on.
- Finally you have the results for each gene family, one gene family per line. In our example above, we are showing the first three gene families, identified by their numbers (the first column, ‘ID’), 8, 10 and 11. The number that appears after a species name in the tree given under ‘Newick’ (e.g., 59 in ‘cat_59’ from gene family 8) is the gene count for that species and that gene family. The third column (‘Family-wide P-value’) tells us for each gene family whether it has a significantly greater rate of evolution. When this value is < 0.01 , then the fourth column (‘Viterbi P-value’) allows the identification of which branches the shift in λ was significant. Because the family-wide p-value of gene families 8 and 10 were not significant in our example, then no results are presented under the Viterbi p-value. However, gene family 11 had a significant p-value (0), and so we can now identify which branches underwent significant contractions or expansions. For this CAFE run, branches 0, 1, 2 and 4 have not undergone significant shifts in λ (as their p-values ≥ 0.05).

4 Troubleshooting

4.1 Known limitations

- Because the random birth and death process assumes that each family has at least one gene at the root of the tree, CAFE will not provide accurate results if included gene families were not present in the most recent common ancestor (MRCA) of all taxa in the tree. For example, even if all taxa have a gene family size of 0, CAFE will assign the MRCA a gene family of size 1, and include the family in estimation of the birth and death rate. This difficulty does not affect analyses containing families that go extinct subsequent to the root node.
- If a change in gene family size is very large on a single branch, CAFE may fail to provide accurate λ estimation and/or die during computation. To see if this is a problem, look at the likelihood scores computed during the λ search (reported in the logfile if the job finishes). If ALL scores are “-inf” then there is a problem with large size change giving CAFE a probability of 0. Removing the family with the largest difference in size among species and rerunning CAFE should allow λ to be estimated on the remaining data. If the problem persists, remove the family with the next largest difference and proceed in a like manner until CAFE no longer finds families with zero probability. However, if rapidly evolving families are removed, care should be taken in interpretation of the estimated average rate of evolution for the remaining data.
- If the product of λ and the distance from the tips to the root is greater than 1, then CAFE will not return accurate results. If λ is specified by the user, this problem is seen as “@@”. If the λ -search option is used, then the value of λ output will be the maximum possible for $\lambda \times t < 1$. If this is a problem, CAFE will print a caution message and “@@” will appear before the Newick-formatted tree in the output. In our experience, this is the most common error encountered by users.
- In very large phylogenetic trees there can be many independent lambda parameters

($2n - 2$ in a rooted tree, where n is the number of taxa). CAFE does not always converge to a single global maximum with large numbers of λ parameters, and therefore can give misleading results. To check for this you should always run the λ search multiple times to ensure that the same estimated values are found. Also, the likelihood of models with more parameters should always be lower than models with fewer parameters, which may not be true if CAFE has failed to find a global maximum. If CAFE does not converge over multiple runs, then one should reduce the number of parameters estimated and try again.

4.2 Looking for help

If you are in need of assistance or want to ask us a question¹, please research the threads in (or post a new thread to) CAFE's [Google group](#).

4.3 License

Copyright © 2010-2017. Licensed by the Trustees of Indiana University under the IU Open Source License, adapted from the Educational Community License, Version 2.0 (ECL-2.0) (<http://opensource.org/licenses/ECL-2.0>). You may not use this file except in compliance with the License. You may obtain a copy of the License from <http://ppa.iu.edu/license>. Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an “AS IS” BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

¹Also, if you have any suggestions to help improve this manual or CAFE's webpage, please write to fkmenides@indiana.edu