User guide for ancestral inference by

Bifurcating Tree with Weighting (BTW) method

Table of Contents

1. Overview

What is ancestral inference by BTW

BTW (Bifurcating Tree with Weighting) is a likelihood-based method to infer ancestral nucleotide states of each site of multiple orthologous genome sequences. This method uses the information of **between species (population) divergence** and **within species (population) polymorphism** for the calculation of the probabilities of ancestral states and allows to infer not only the ancestral states between species but also **those of polymorphic sites within species**.

The biggest difficulty of ancestral inference using multiple genomes within species is that each site in these genomes can have different genealogy shape by recombination. This makes difficult to calculate the likelihood of each nucleotide substitution scenario assuming all sites in the genomes share a single tree as done in the previous ancestral inference methods like BASEML in PAML (Yang 2007). To overcome this difficulty, BTW method first compresses the information of within species polymorphism into two artificially made nucleotide sequences, they are called as "collapse sequences". For these two collapse sequences, a bifurcating tree is the only one possible tree and we can apply ancestral inference by BASEML assuming all sites in the two collapse sequences share the bifurcating tree with genomes from outgroup to root the bifurcating tree. The inferred ancestral states can be considered as those of polymorphic sites within species. Finally, BTW method weights the estimated probabilities of ancestral states by BASEML based on an expected shape of site frequency spectrum (SFS) to incorporate the information of within species polymorphism which could not be compressed into the collapse sequences.

The idea of BTW method was proposed in Matsumoto and Akashi (2018) and its accuracy and precision in a study of base composition evolution were also evaluated in the same paper. In this document, we would like to provide a practical instruction of how to run ancestral inference by BTW method using the set of codes provided in Yamashita *et al*. (2025), which have been confirmed to run on Mac OSX (High Sierra ~ Sonoma).

Flow of ancestral inference by BTW

BTW method can be divided into the three processes;

(1) make the collapse sequences

(2) run BASEML using the collapse sequences and outgroup genomes

(3) weight the estimated probabilities of ancestral states by BASEML based on an expected SFS.

In the below sections, we will explain about each process and the required inputs of the codes to run it and finally, we will explain about the code which concatenates the explained codes and run the whole processes of BTW method.

***The descriptions about the codes and inputs that the users may have to change to analyze their own data are highlighted in blue.**

2. Make collapse sequences

Process to make collapse sequences

The first process of BTW method is to make two collapse sequences for each species in which ≥ 3 genomes to be analyzed. The two collapse sequences show the two information of within species polymorphism, **(1) whether a site is polymorphic or not, and (2) what kind of nucleotides are there**.

A site in the aligned genomes is checked whether it is monomorphic or polymorphic within species. If a site is monomorphic with nucleotide *X*, the two collapse sequences also have nucleotide *X* at the same site. If a site is polymorphic with nucleotide *Y* and *Z*, one of the two collapse sequence has nucleotide *Y* and the other has nucleotide *Z* at the same site. Which collapse sequence has nucleotide *Y* is decided randomly. A polymorphic site with more than two states cannot be used in the collapse sequences. Repeating these processes from the first to the last site of the aligned genomes, the two collapse sequences of one species are made. For the graphical explanation, please see Figure 2 of Matsumoto and Akashi (2018).

Codes and input data/parameters

We provide a perl code **"make_collapse_seqs_test_nucleotide.pl**" which reads input alignment file and parameters and outputs the collapse sequences. The input parameters are written in a shell script "**run_make_collapse_seqs.sh**", which passes the inputs to the perl code and runs it. They are in the "**____codes**" directory.

For the format of the genome alignment file, please also check the examples we provided. **The perl code cannot consider "non-TCAG states" in the nucleotide sequences like gap or N, so please filter them beforehand. Also please filter polymorphic sites with more than two states within species**. When the perl code recognizes these polymorphic sites, it outputs error message but does not stop and outputs incorrect collapse sequences. **A genome of one sample should be given as ">name" in one line and "nucleotide sequence" in the next single line of the file. Also please add an empty line at the end of the file as below.**

```
>sample1_speciesA
ATGACGCGCTACAAGCAGGAATTCACGGAGGACGACTCGAGTTCCATCGGCGGCATTCAA
>sample2_speciesA
ATGACGCGCTACAAGCAGGAATTCACGGAGGACGACTCGAGTTCCATCGGCGGCATTCAA
>sample3_speciesA
ATGACGCGCTACAAGCAGGAATTCACGGAGGACGACTCGAGTTCCATCGGCGGCATTCAA
>sample4_speciesA
ATGACGCGCTACAAGCAGGAATTCACGGAGGACGACTCGAGTTCCATCGGCGGCATTCAA
>outgroup_speciesB
ATGACACGCTACAAGCAGGAATTCACTGAAGACGACTCCAGTTCCATCGGCGGCATTCAA
(empty line)
```

The perl code reads the alignment file and the input parameters written in the shell script. **The number of input parameters changes depending on the number of species in the input alignment file to be analyzed.** Here we would like to explain based on one of our example (in **"__example1/"**).

Input parameters in **run_make_collapse_seqs.sh**

a: number of species for which the collapse sequences will be made

(In the example, the collapse sequences are made for each of the two species, *Dmel* and *Dsim* (name species 0 and species1, respectively), so a = 2)

b0: order of the first sequence of species 0 in the alignment file

(In the example, 1st to 10th genomes are from species 0, so b0 = 1)

c0: order of the last sequence of species 0 in the alignment file

(In the example, 1st to 10th genomes are from species 0, so c0 = 10)

b1: order of the first sequence of species 1 in the alignment file

(In the example, 11th to 20th genomes are from species 1, so b1 = 11)

c1: order of the last sequence of species 1 in the alignment file

(In the example, 11th to 20th genomes are from species 1, so c1 = 20)

d: number of species for which a single genome will be used

(In the example, there are two species with a single genome, *Dyak* and *Dere* (name single_seq_species 0 and single_seq_species 1, respectively), so d = 2)

e0: order of the single genome of single_seq_species 0 in the alignment file

(In the example, 21th genome is from single_seq_species 0, so e0 = 21)

e1: order of the single genome of single_seq_species 1 in the alignment file

(In the example, 22th genome is from single_seq_species 1, so e1 = 22)

***Because of the parameter setting as above, there is one more requirement in the format of the input alignment file. Genomes from one species cannot be scattered and should be contiguously listed from their first to last sequences in the input alignment file.**

These parameters are passed to the perl code as

perl make_collapse_seqs_test_nucleotide.pl $a $b0 $c0 $b1 $c1 $d $e0 $e1

in the shell script.

It is possible to increase or decrease the input parameters depending on the number of species. **If a = *n* and d = *m* in the input alignment file, there will be b0 ~ b(*n*-1), c0 ~ c(*n*-1) and e0 ~ e(*m*-1) parameters**. For our another example (in **"__example2/"**),

a=6 (Collapse sequences for *Dmel*, *Dsim*, *Dtei*, *Dyak*, *Dere* and *Dore*)

b0=1 (The 1st genome in the alignment file is the first sequence of *Dmel*)

c0=10 (The 10th genome in the alignment file is the last sequence of *Dmel*)

b1=11 (The 11th genome in the alignment file is the first sequence of *Dsim*)

c1=20 (The 20th genome in the alignment file is the last sequence of *Dsim*)

b2=21 (The 21th genome in the alignment file is the first sequence of *Dtei*)

c2=30 (The 30th genome in the alignment file is the last sequence of *Dtei*)

b3=31 (The 31th genome in the alignment file is the first sequence of *Dyak*)

c3=40 (The 40th genome in the alignment file is the last sequence of *Dyak*)

b4=41 (The 41th genome in the alignment file is the first sequence of *Dere*)

c4=50 (The 50th genome in the alignment file is the last sequence of *Dere*)

b5=51 (The 51th genome in the alignment file is the first sequence of *Dore*)

c5=60 (The 60th genome in the alignment file is the last sequence of *Dore*)

d=0 (There is no species with a single genome)

and rewrite the shell script as

perl make_collapse_seqs_test_nucleotide.pl $a $b0 $c0 $b1 $c1 $b2 $c2 $b3 $c3 $b4 $c4 $b5 $c5 $d

The output of the codes is the alignments of the collapse sequences and single genomes with the same format to the input alignment file. **The order of the nucleotide sequences in the output alignment file when a = *n* and d = *m* is**

single genome of single_seq_species 0

single genome of single_seq_species 1

.

.

.

single genome of single_seq_species *m*

first collapse sequence of species 0

second collapse sequence of species 0

first collapse sequence of species 1

second collapse sequence of species 1

.

.

.

first collapse sequence of species *n*

second collapse sequence of species *n*

The order of the nucleotide sequences in the output alignment file will be used to make an input tree file to run BASEML in the Section 3.

3. Run BASEML in PAML package

Overview of BASEML

BASEML is a software implemented in PAML (Yang 2007) to estimate the probabilities of ancestral nucleotide states of given nucleotide sequences at each node of a given tree. BASEML can take several kinds of nucleotide substitution model for the ancestral inference and have shown to show high accuracy and precision to study evolution of base composition (Matsumoto *et al*. 2015). The detailed instruction of PAML (including BASEML) has provided by Dr. Yang.
http://abacus.gene.ucl.ac.uk/software/pamlDOC.pdf

The second step of the ancestral inference by BTW method is run this BASEML using the alignment file of the collapse sequences and single genomes made above and a tree file for these sequences. As explained, assuming bifurcating trees to the two collapse sequences from one species and a reliable species phylogeny to the genomes from different species allow to assign a single tree to the nucleotide sequences to be analyzed and conduct ancestral inference by BASEML.

BASEML under GTR-NH$_b$ substitution model

BASEML has an option to choose one nucleotide substitution model to be used for the ancestral inference. In BTW method, a model called **GTR-NH$_b$ is recommended and set as the default model to be used**. This model assumes that nucleotide substitutions have occurred based on GTR model (Tavare 1986, Yang 1994 and Zharkikh 1994) allowing each of the eight parameters in the model be different among lineages. This feature makes GTR-NH$_b$ model to have high generality and show high accuracy and precision in ancestral inference even if the base composition is non-stationary changing in the lineages and substitution rates are heterogeneous among the lineages. For the more detailed information of GTR-NH$_b$ model, please check Matsumoto *et al*. (2015).
The input parameters for the nucleotide substitution model are written in **"_AI_for_collapse/2_BASEML_and_HM/02_ctlfiles/sample_ctl/5_Baseml_Ctl/model3.ctl"**. The parameter values for GTR-NH$_b$ model are

model = 7

fix_kappa = 2

nhomo = 4 or 5 (difference is explained below)

To use other models, please follow the instruction of PAML.

Tree file for alignment file with collapse sequences

Another important input of BASEML is the topological relationship of the input nucleotide sequences. In our package, it is written in **"_AI_for_collapse/2_BASEML_and_HM/03_seq_dat/sample.trees"**.

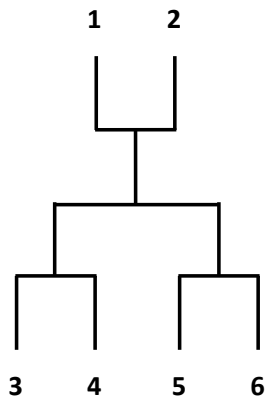
For the basic format of a tree topology, please follow the instruction of PAML. For BTW method, the two collapse sequences of one species should be the closest and the topology between species should follow a species phylogeny (please also check the Figure 2 of Matsumoto and Akashi 2018). In addition, **using an option to assume the shared parameter values of GTR-NH$_b$ model between the two lineages of the two collapse sequences is recommended**. **This option is available by setting nhomo = 5 by which the lineages sharing the parameter values can be designated. If nhomo = 4 is used, all lineages in the tree are assumed to be able to have different parameter values each other.**

In our example, the tree topology is given as

6 1

((1 #0, 2 #1) #4, (3 #2, 4 #2) #5, (5 #3, 6 #3) #6) #7;

which means there is one tree with six nucleotide sequences with a topology as below.



The "number (index)" assigned to each nucleotide sequence has to be consistent with the order of the nucleotide sequences in the input alignment file of BASEML. In our example, the order is

single genome of single_seq_species 0 (*Dyak*)

single genome of single_seq_species 1 (*Dere*)

first collapse sequence of species 0 (*Dmel*)

second collapse sequence of species 0 (*Dmel*)

first collapse sequence of species 1 (*Dsim*)

second collapse sequence of species 1 (*Dsim*)

Therefore, nucleotide sequence "1" is the single genome of *Dyak*, "2" is the single genome of *Dere*, "3" and "4" are the collapse sequences of *Dmel*, and "5" and "6" are the collapse sequences of *Dsim*.

The lineages assigned the same "#X" are assumed to share the parameter values. So in our example, external lineages of the nucleotide sequences "3" and "4", and those of "5" and "6" are assumed to share the parameter values, respectively because they are the lineages of the collapse sequences. **The "#X" in the tree is specific to the case of nhomo = 5. If nhomo = 4, tree without "#X" can be used**.

For a given set of the input alignment file and tree topology, BTW method replicates BASEML 10 times and choose one replicate showed the highest likelihood as recommended in Matsumoto *et al*. (2015) as the most accurate result among the replicates.

4. Weight ancestral probabilities by the expected site frequency spectrum

As explained above, the two collapse sequences of one species contain the information of (1) whether a site is polymorphic or not, and (2) what kind of nucleotides are there. However, the information of the **frequency of each polymorphic mutation within species** was ignored and not incorporated in the result of the ancestral inference by BASEML. Although the ancestral states between the two collapse sequences inferred by BASEML can be considered as those of polymorphic sites within species, their accuracy may not be very high because of this ignorance (Bifurcating Tree (BT) method in Matsumoto and Akashi 2018). The last process of BTW method is to weight the estimated ancestral probabilities by BASEML to incorporate the information of the frequency of each polymorphic mutation within species and increases the accuracy of ancestral inference.

Process of the weighting

Consider that at one site of the genomes, nine of the ten samples show nucleotide "A" (frequency is nine) and one shows nucleotide "G" (frequency is one). In such a case, **if the expected shape of the SFS of the mutations on this site is given, it is possible to estimate the probability that nucleotide "A" (or "G") is the ancestral state**. For example, if the expected SFS is that of selectively neutral mutations at the equilibrium (SFS_{ne}), the expected proportions of derived mutation with frequency one and nine are 0.353… and 0.039…, respectively (Fisher 1930, Wright 1969). This means that a derived mutation is expected to be observed as a singleton (frequency is one) with nine times higher probability than as a derived mutation with frequency nine. Therefore, the probability that a nucleotide with frequency nine is the ancestral state is nine times higher than that a nucleotide with frequency one is the ancestral state. In BTW method, these probabilities of the ancestral states are multiplied to the probabilities estimated by BASEML. For the example site above, if BASEML estimated that "A" and "G" is the ancestral state with probability 0.8 and 0.2, respectively and if we assume SFS_{ne} as the expected SFS, the probabilities of that "A" and "G" is the ancestral state after the weighting is $0.8\times0.9=0.72$ and $0.2\times0.1=0.02$, respectively. After normalizing the probabilities to make the sum in one site to be 1.0, they become 0.972… and 0.027…, respectively. BTW method does this weighting

process for each site based on the assumed expected SFS and recalculate the probabilities of the ancestral states estimated by BASEML.

Iterative BTW_{est}

Matsumoto and Akashi (2018) found that if the expected SFS assumed for the weighting and the SFS estimated from the weighted probabilities are compared, the latter is closer to the actual SFS. Based on this result, Matsumoto and Akashi (2018) proposed an approach of ancestral inference by BTW method, which is named iterative BTW_{est}.

In iterative BTW_{est}, first the estimated probabilities of the ancestral states by BASEML are weighted assuming SFS_{ne} as the expected SFS. Then, from the weighted probabilities of the ancestral state by BTW, SFS in the analyzed genomes is estimated. Then, the weighting is done again using this estimated SFS as the expected and estimate the new SFS from the weighted probabilities (first iteration). In the second iteration, the estimated SFS in the first iteration is used as the expected SFS for the weighting and new SFS is estimated from the weighted probabilities. The iteration is replicated until the newly estimated SFS and the expected SFS used for the weighting become almost the equal. Iterative BTW_{est} approach allows to apply BTW method even if the assumption of SFS_{ne} as the expected SFS is unrealistic.

Codes and input data/parameters

The required input data to run iterative BTW_{est} is **(1) the alignment file of the original genome sequence** ("original_alignment" which was the input to make the collapse sequences), **(2) the alignment file of the collapse sequences and single genomes** ("collapse_alignment") and **(3) "fltrst" file** which is the output of BASEML. The input parameters are written in **"_run_iterative_BTWest.sh"** in the "____codes" directory. Again, we would like to explain based on our example (in **"__example1/"**).

Input parameters in **"_run_iterative_BTWest.sh"**

a: number of the nucleotide sequences in the collapse_alignment

(In the example, there are two single genomes from *Dyak*, *Dere* and two collapse sequences from each of *Dmel* and *Dsim*, so a = 6)

b: number of internal nodes in the input tree of BASEML

(In the example, there are four internal nodes in the tree (please check the figure shown in the section 3), so b = 4)

c: number of the species for which the collapse sequences were made

(In the example, the collapse sequences were made for *Dmel* and *Dsim* (name species 0 and species1, respectively), so c = 2)

d0: order of the first genome of species 0 in the original_alignment

(In the example, 1st to 10th genomes are from species 0, so d0 = 1)

e0: order of the last genome of species 0 in the original_alignment

(In the example, 1st to 10th genomes are from species 0, so e0 = 10)

d1: order of the first genome of species 1 in the original_alignment

(In the example, 11th to 20th genomes are from species 1, so d1 = 11)

e1: order of the last genome of species 1 in the original_alignment

(In the example, 11th to 20th genomes are from species 1, so e1 = 20)

f0: order of the first collapse sequence of species 0 in collapse_alignment

(In the example, 3rd and 4th nucleotide sequences are the collapse sequences of species 0, so f0 = 3)

g0: order of the second collapse sequence of species 0 in the collapse_alignment

(In the example, 3rd and 4th nucleotide sequences are the collapse sequences of species 0, so g0 = 4)

f1: order of the first collapse sequence of species 1 in the collapse_alignment

(In the example, 5th and 6th nucleotide sequences are the collapse sequences of species 1, so f1 = 5)

g1: order of the second collapse sequence of species 1 in the collapse_alignment

(In the example, 5th and 6th nucleotide sequences are the collapse sequences of species 1, so g1 = 6)

i: number of the category of the mutations sharing the same expected SFS

(In the example, i=12 is assigned which means the SFS of each of the 12 mutations between four nucleotides are estimated separately and used for the weighting.

If some of the 12 mutations are assumed to share the same expected SFS, please change i from 1 (all mutations share the same expected SFS) to 12 (no sharing of the expected SFS)).

mut_TC ~ mut_GA: categories of each of the 12 mutations. mutations with the same category are assumed to share the same expected SFS.

(In the example, different value from 1 to 12 is assigned for each of the 12 mutations because i = 12. For the mutations having the same parameter values, the estimated SFS are pooled to estimate the expected SFS of the category.)

iteration: number of the iteration in iterative BTWest

(In the example, the number of the iteration is 5))

It is possible to increase or decrease the input parameters depending on the number of species. **If c = *n* in the data, there will be d0 ~ d(*n*-1), e0 ~ e(*n*-1), f0 ~ f(*n*-1) and g0 ~ g(*n*-1) parameters. Please check the parameter setting in the shell script in "__example2/" for another example.**

Output of the code

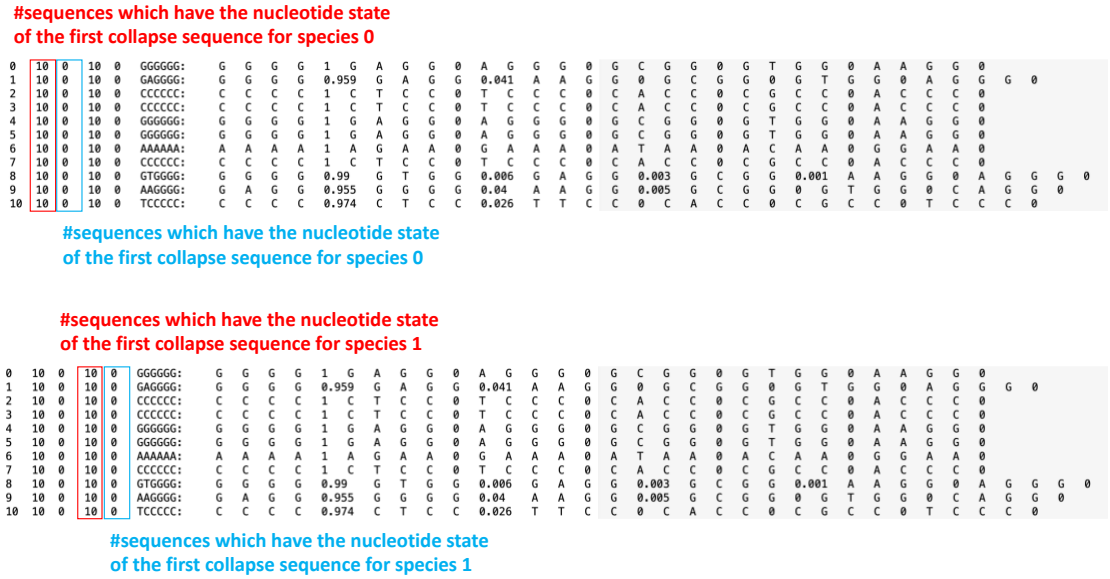
The output of the BTW method is **"anc_site_probs_all_node.txt"** and is stored in the directory **"iterative_BTWest/"**.

The first column of this file shows the position of the site of the original_alignment.

site position in the original_alignment

```
0   10  0   10  0   GGGGGG:   G  G  G  G  1      G  A  G  G  0      A  G  G  G  0      G  C  G  G  0      G  T  G  G  0  A  A  G  G  0
1   10  0   10  0   GAGGGG:   G  G  G  G  0.959  G  A  G  G  0.041  A  A  G  G  0      G  C  G  G  0      G  T  G  G  0  A  G  G  G  0
2   10  0   10  0   CCCCCC:   C  C  C  C  1      C  T  C  C  0      T  C  C  C  0      C  A  C  C  0      C  G  C  C  0  A  C  C  C  0
3   10  0   10  0   CCCCCC:   C  C  C  C  1      C  T  C  C  0      T  C  C  C  0      C  A  C  C  0      C  G  C  C  0  A  C  C  C  0
4   10  0   10  0   GGGGGG:   G  G  G  G  1      G  A  G  G  0      A  G  G  G  0      G  C  G  G  0      G  T  G  G  0  A  A  G  G  0
5   10  0   10  0   GGGGGG:   G  G  G  G  1      G  A  G  G  0      A  G  G  G  0      G  C  G  G  0      G  T  G  G  0  A  A  G  G  0
6   10  0   10  0   AAAAAA:   A  A  A  A  1      A  G  A  A  0      G  A  A  A  0      A  T  A  A  0      A  C  A  A  0  G  G  A  A  0
7   10  0   10  0   CCCCCC:   C  C  C  C  1      C  T  C  C  0      T  C  C  C  0      C  A  C  C  0      C  G  C  C  0  A  C  C  C  0
8   10  0   10  0   GTGGGG:   G  G  G  G  0.99   G  T  G  G  0.006  G  A  G  G  0.003  G  C  G  G  0.001  A  A  G  G  0  A  G  G  G  0
9   10  0   10  0   AAGGGG:   G  A  G  G  0.955  G  G  G  G  0.04   A  A  G  G  0.005  G  C  G  G  0      G  T  G  G  0  C  A  G  G  0
10  10  0   10  0   TCCCCC:   C  C  C  C  0.974  C  T  C  C  0.026  T  T  C  C  0      C  A  C  C  0      C  G  C  C  0  T  C  C  C  0
```

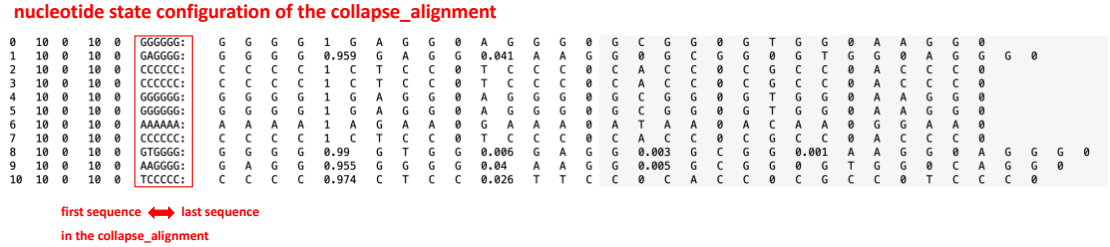
The next columns show **the frequencies of the nucleotide states in the original_alignment for each species**. Nucleotide states in the original_alignment should be reflected in the two states in the collapse sequences. These columns show (1) the number of genomes which has the nucleotide state exists in the first collapse sequence and (2) the number of genomes which has the nucleotide state exists in the second collapse sequence.



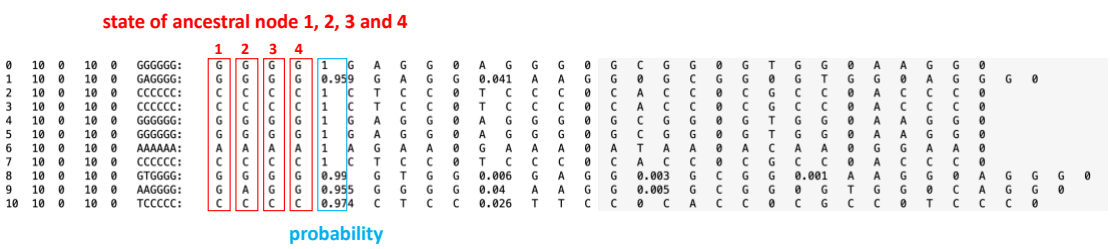
The pattern "*n* 0" means the site is monomorphic (*n* is the total number of genomes of the species).

The number of the columns changes depending on the number of species to which the collapse sequences were made.

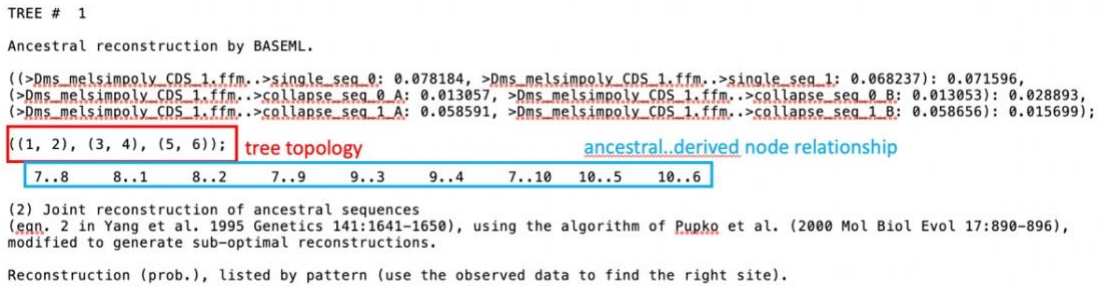
The next column shows **the nucleotide states configuration of the collapse_alignment**. The nucleotide state of each site of each nucleotide sequence is listed. The order is from the first to the last nucleotide sequences in the collapse_alignment. In our example (in "**__example1/**"), the order becomes *Dyak*, *Dere*, *Dmel*_collapse1, *Dmel*_collaspe2, *Dsim*_collapse1 and *Dsim*_collapse2.



After the column of the nucleotide states configuration of the collapse_alingment, **"anc_site_probs_all_node.txt" lists the nucleotide states configuration of each ancestral node of the tree and its estimated probability**.



To understand the position of each ancestral node in the tree, it is helpful to check "fltrst" file (in the "BASEML_result" directory)**.**



The tree topology in this file shows that the input collapse_alignment contained six nucleotide sequences named 1~6 (in the example, they are the single genomes of *Dyak* and *Dere*, and the collapse sequences of *Dmel* and *Dsim*. Please also check the tree in the Section 3). The **"ancestral..derived node relationship" below shows which of the two nodes are connected in the tree**. In the example, the single genomes of *Dyak* and *Dere* are connected at node 8, the collapse sequences of *Dmel* and *Dsim* are connected at node 9 and 10, respectively, and nodes 8, 9 and 10 are connected at node 7 (root). **Because nodes 1~6 are the six nucleotide sequences in the collapse_alignment, nodes 7~10 are renamed as ancestral node 1~4, and their states are listed in the "anc_site_probs_all_node.txt"**.

As explained, the ancestral state between the two collapse sequences can be considered as that of the within species polymorphism. Therefore, from the information shown in "anc_site_probs_all_node.txt", it is possible to polarize the polymorphism observed in the original_alingment. Also, by comparing the inferred states of the two ancestral nodes, it is possible to infer fixations occurred between these two nodes.

5. Run the concatenated code to do the whole processes

Finally, we provide a concatenated code to run the whole processes explained above. To run the code, please execute the shell script **"run_BTW_pipeline.sh"** in the terminal window.

./ run_BTW_pipeline.sh

What the users have to prepare is **(1) the original genome alignments and file list in "_seq_folder" directory, (2) file list to specify the alignment files to be concatenated in "_seq_list_to_be_concatenated" directory and (3) tree file of collapse_alignment in "_tree_folder" directory**. Please make sure the alignment and tree files are formatted as explained in the Section 2 and 3. **Also, the settings of the input parameters in "__run_iterated_BTWest.sh" and "__run_make_collapse_seqs" have to be done by the users depending on their input data**.

This pipeline outputs the collapse sequences in **"_collapse_seqs_folder"** directory, and concatenated original genome alignment in **"_concatenated_original_alignment_folder"** directory.

The results of the BASEML and iterative BTW are in **"BASEML_result"** and **"iterative_BTWest"** directories, respectively.

To run the examples, first please copy **"__run_iterated_BTWest.sh"** and **"__run_make_collapse_seqs"** in the example directory to **"____codes"** directory. Then, please copy **"_seqs_folder"**, **"_seq_list_to_be_concatenated"** and **"_tree_folder"** directories in the example directory to **"BTW_pipeline"** directory. Finally, please execute **"run_BTW_pipeline.sh"**.

References

Fisher, R. A., 1930 The Genetical Theory of Natural Selection. Clarendon Press, Oxford.

Matsumoto, T. and H. Akashi, 2018 Distinguishing among evolutionary forces acting on genome-wide base composition: Computer simulation analysis of approximate methods for inferring site frequency spectra of derived mutations in recombining regions. G3 8: 1755-1769.

Matsumoto, T., H. Akashi, and Z. Yang, 2015 Evaluation of Ancestral Sequence Reconstruction Methods to Infer Nonstationary Patterns of Nucleotide Substitution. Genetics 200: 873–890.

Tavaré S, 1986 Some probabilistic and statistical problems on the analysis of DNA sequences. Lect Math Life Sci 17:57-86.

Wright, S, 1969 Evolution and the Genetics of Populations. Vol. 2 The theory of Gene Frequencies. University of Chicago Press, Chicago.

Yang Z, 1994 Estimating the pattern of nucleotide substitution. J Mol Evol 39:105-111.

Yang Z, 2007 PAML 4: Phylogenetic analysis by maximum likelihood. Mol Biol Evol 24:1586-1591.

Zharkikh A, 1994 Estimation of evolutionary distances between nucleotide sequences. J Mol Evol 39:315-329.

Yamashita, H., T. Matsumoto, K. Kawashima, H. S. Abdulla Daanaa, Z. Yang, and H. Akashi, 2025 Dinucleotide preferences underlie apparent codon preference reversals in the *Drosophila melanogaster* lineage. PNAS 122(21) e2419696122.