InferNetwork_MPL

Description

Infers a species network(s) with a specified number of reticulation nodes using maximum pseudo-likelihood. The returned species network(s) will have inferred branch lengths and inheritance probabilities. During the search, branch lengths and inheritance probabilities of a proposed species network can be either sampled or optimized. For the first case, after the search, branch lengths and inheritance probabilities of all top species networks are optimized before being returned. In order to address the identifiability issue caused by the fact that networks are not necessarily encoded by their triplet system, users can ask the program to further optimize those parameters of the inferred network under full likelihood. We use Richard Brent's algorithm (from his book "Algorithms for Minimization without Derivatives", p. 79) to optimize the branch lengths and inheritance probabilities to obtain the maximum (pseudo-) likelihood for that species network. The species network and gene trees must be specified in the Rich Newick Format.

The inference uses only topologies of gene trees. The input gene trees can be non-binary and can contain missing taxa. The input gene trees can be gene tree distributions inferred from Bayesian methods like MrBayes. See the second example below.

Usage

```
InferNetwork_MPL geneTreeList numReticulations [-a taxa map] [-b threshold] [-s startingNetwork] [-fs] [-n
numNetReturned] [-h {s1 [,s2...]}] [-w (w1,...,w7)] [-x numRuns] [-m maxNetExamined] [-md moveDiameter] [-rd
reticulationDiameter] [-f maxFailure] [-o] [-po] [-p (rel,abs)] [-r maxRounds] [-t maxTryPerBr] [-i
improveThreshold] [-l maxBL] [-pl numProcessors] [-di] [resultOutputFile]
```

geneTreeList	Comma delimited list of gene tree identifiers or comma delimited list of sets of gene tree identifiers. See details.	mandatory
numReticulations	Maximum number of reticulations to added.	mandatory
-b *threshold*	Gene trees bootstrap threshold. Edges in the gene trees that have support lower than *threshold* will be contracted.	optional
-a *taxa map*	Gene tree / species tree taxa association.	optional
-s *startingNetwork*	Specify the network to start search. Default value is the optimal MDC tree.	optional
-fs	Fix the start tree for search. If specified and give a start tree (-s), the search will fix the topology of the start tree.	optional
-n *numNetReturned*	Number of optimal networks to return. Default value is 5.	optional
-h *{s1 [, s2...]}*	A set of specified hybrid species.	optional
-w *(w1, ..., w7)*	The weights of operations for network arrangement during the network search. Default value is (0.1,0.1,0.15,0.55,0.15,0.15,2.8).	optional
-x *numRuns*	The number of runs of the search. Default value is 10.	optional
-m *maxNetExamined*	Maximum number of network topologies to examined. Default value is infinity.	optional
-md *moveDiameter*	Maximum diameter to make an arrangement during network search. Default value is infinity.	optional
-rd *reticulationDiameter*	Maximum diameter for a reticulation event (the distance between two parents of a reticulation node). Default value is infinity.	optional
-f *maxFailure*	Maximum consecutive number of failures for hill climbing. Default value is 100.	optional
-o	If specified, during the search, for every proposed species network, its branch lengths and inheritance probabilities will be optimized to compute its likelihood. Default value is false.	optional
-po	If specified, after the search the returned species networks will be optimized for their branch lengths and inheritance probabilities under full likelihood. Default value is false.	optional
-p *(rel, abs)*	The original stopping criterion of Brent's algorithm. Default value is (0.01, 0.001).	optional

-r *maxRound*	Maximum number of rounds to optimize branch lengths for a network topology. Default value is 100.	optional
-t *maxTryPerBr*	Maximum number of trial per branch in one round to optimize branch lengths for a network topology. Default value is 100.	optional
-i *improveThreshold*	Minimum threshold of improvement to continue the next round of optimization of branch lengths. Default value is 0.001.	optional
-l *maxBL*	Maximum branch lengths considered. Default value is 6.	optional
-pl *numProcessors*	Number of processors if you want the computation to be done in parallel. Default value is 1.	optional
-di	Output the Rich Newick string of the inferred network that can be read by Dendroscope .	optional
resultOutputFile	Optional file destination for command output.	optional

It is mandatory to specify the number of reticulation nodes to added to the starting network. By default, it is assumed that only one individual is sampled per species in gene trees. However, the option `[-a taxa map]` allows multiple alleles to be sampled. If users have a prior knowledge of the hybrid species, they can specify them using option *-h*.

The search: Option *-m* allows users to specify the maximum number of networks examined during the search. Users can specify the weights of seven operations for network arrangement through option *-w*. The seven weights correspond to adding a reticulation node, deleting a reticulation node, relocating the head of a reticulation edge, relocating the tail of an edge, reversing the direction of a reticulation edge, replacing a reticulation edge and changing branch lengths and inheritance probabilities, respectively. Furthermore, users can use option *-md* to specify the maximum move diameter of an operation for network rearrangement, like what local-SPR does. Also, users can use option *-rd* to specify the maximum reticulation diameter which is the distance (the number of branches) between the two parents of a reticulation node. In order to avoid getting stuck at some local optimum, it is recommended to performed the search multiple times, which users can specify by option *-x*. The *-s* option allows the users to specify a starting network (can be a tree) for network search. If the starting network is not specified, the optimal tree under MDC (command infer_ST_MDC) will be used. If it is not binary, a random resolution will be used. By default, only the first optimal species network will be returned. However, users can use *-n* option to ask for multiple optimal networks.

During the search, by default, simulated annealing is used (See *Salter* and *Pearl* 2001 for details of settings), where the branch lengths and inheritance probabilities are sampled. In this case, through option *-po,* as a post-processing, users can optimize the branch lengths and inheritance probabilities of the species networks returned by the search. If the dataset is not large and a large a mount of memory is available, users can use option *-o* to optimize the branch lengths and inheritance probabilities of every proposed network during the search. In this case, simple hill climbing will be used, and only the first 5 operations for network arrangement will be used.

By default, the method returns the species networks inferred under maximum pseudo-likelihood. However, networks are not necessarily encoded by their triplet system. To address this issue, users can ask PhyloNet to further analyze those returned species networks by optimizing their branch lengths and inheritance probabilities under full likelihood using option *-o.* Keep in mind that this feature might not be feasible for large datasets.

We use Richard Brent's algorithm (from his book "Algorithms for Minimization without Derivatives", p. 79) to optimize the branch lengths and inheritance probabilities. Users can use different options to control this process. Option *-p* allows users to specify the original stopping criterion of Brent's algorithm. More precisely, *abs* and *rel* define a tolerance $tol = rel\,|x| + abs$. We optimize the branch lengths one by one. For every branch, it terminates when either *maxTryPerBr* (option *-t*) trials have been made or the Brent's algorithm suggests so. Users can put an upper bound of the branch lengths through option *-l*. Optimization of all branch lengths consists of a round. After every round, if the improvement in terms of likelihood score is greater than that from last round by at least *improveThreshold* (option *-i*), we starts next round. A maximum of *maxRound* (option *-r*) rounds will be tried.

If users want to run the computation in parallel (in terms of the gene trees). Please specify the number of processors through option *-pl*.

Examples

```
#NEXUS

BEGIN TREES;

Tree gt0=(a:2.299,((e:1.329,f:1.329):0.77,(c:1.684,(b:1.232,d:1.232):0.451):0.416):0.2);
Tree gt1=(a:2.085,((e:1.376,f:1.376):0.696,(d:1.487,(b:1.241,c:1.241):0.246):0.585):0.013);
Tree gt2=(((c:0.52,d:0.52):1.243,(e:1.403,f:1.403):0.36):1.025,(a:1.863,b:1.863):0.925);
Tree gt3=(a:2.82,((b:1.051,(c:0.86,d:0.86):0.19):1.357,(e:1.365,f:1.365):1.043):0.412);
Tree gt4=((e:1.3,f:1.3):0.994,(a:1.869,(b:1.255,(c:0.849,d:0.849):0.405):0.615):0.425);
Tree gt5=(a:2.46,((b:1.077,c:1.077):0.857,(f:1.141,(d:0.505,e:0.505):0.636):0.793):0.526);
Tree gt6=(a:2.025,((b:1.111,c:1.111):0.416,(f:1.304,(d:0.727,e:0.727):0.577):0.223):0.498);
Tree gt7=((d:1.526,(e:1.415,f:1.415):0.111):0.982,(a:2.188,(b:1.532,c:1.532):0.656):0.32);
Tree gt8=(a:2.234,((b:1.057,c:1.057):0.766,(f:1.301,(d:0.849,e:0.849):0.452):0.522):0.411);
Tree gt9=((e:1.644,(b:1.361,(c:0.503,d:0.503):0.858):0.283):0.787,(a:2.226,f:2.226):0.205);
Tree gt10=(a:2.917,((e:1.683,(c:0.961,d:0.961):0.722):0.886,(b:1.779,f:1.779):0.79):0.348);
Tree gt11=(a:2.391,((b:1.041,(c:0.602,d:0.602):0.439):0.516,(e:1.164,f:1.164):0.393):0.834);
Tree gt12=((b:1.21,c:1.21):1.622,(a:2.443,(f:1.804,(d:0.583,e:0.583):1.221):0.639):0.389);
Tree gt13=(a:2.047,((b:1.025,c:1.025):0.519,(f:1.295,(d:0.738,e:0.738):0.556):0.249):0.503);
Tree gt14=(a:2.58,((d:0.919,e:0.919):0.834,(f:1.503,(b:1.228,c:1.228):0.275):0.251):0.827);
Tree gt15=((f:1.267,(d:0.871,e:0.871):0.396):1.67,(a:2.181,(b:1.362,c:1.362):0.819):0.756);
Tree gt16=(a:3.016,(b:1.892,(c:1.816,(f:1.479,(d:0.812,e:0.812):0.667):0.337):0.076):1.124);
Tree gt17=((f:1.186,(d:0.721,e:0.721):0.465):1.822,(a:2.031,(b:1.13,c:1.13):0.902):0.977);
Tree gt18=((c:1.51,(f:1.166,(d:0.521,e:0.521):0.645):0.345):1.218,(a:2.073,b:2.073):0.655);
Tree gt19=(a:2.329,((b:1.354,c:1.354):0.467,(f:1.392,(d:0.955,e:0.955):0.437):0.429):0.508);
Tree gt20=(a:3.31,((e:1.083,f:1.083):2.08,(b:1.923,(c:0.538,d:0.538):1.385):1.241):0.146);
Tree gt21=((e:1.271,f:1.271):0.969,(a:1.687,(c:1.131,(b:1.012,d:1.012):0.119):0.556):0.554);
Tree gt22=(c:4.001,((f:1.146,(d:0.896,e:0.896):0.25):1.216,(a:1.542,b:1.542):0.82):1.639);
Tree gt23=((e:1.147,f:1.147):1.081,(a:1.658,(b:1.08,(c:0.611,d:0.611):0.469):0.578):0.569);
Tree gt24=(f:3.318,(a:2.154,((b:1.086,c:1.086):0.78,(d:1.105,e:1.105):0.761):0.287):1.164);
Tree gt25=(a:2.153,((b:1.031,c:1.031):0.598,(f:1.469,(d:0.638,e:0.638):0.831):0.16):0.524);
Tree gt26=(a:4.225,((b:1.1,c:1.1):0.808,(f:1.365,(d:0.7,e:0.7):0.665):0.543):2.316);
Tree gt27=((a:1.756,b:1.756):1.852,(c:2.162,(d:1.829,(e:1.149,f:1.149):0.68):0.333):1.446);
Tree gt28=(a:3.675,((f:1.002,(d:0.652,e:0.652):0.35):0.898,(b:1.342,c:1.342):0.558):1.775);
Tree gt29=((b:1.343,c:1.343):0.843,(a:2.014,(e:1.686,(d:1.523,f:1.523):0.163):0.328):0.171);
Tree gt30=(a:2.171,((b:1.223,c:1.223):0.631,(f:1.63,(d:0.528,e:0.528):1.101):0.224):0.317);
Tree gt31=(a:2.022,((b:1.134,c:1.134):0.877,(f:1.147,(d:0.891,e:0.891):0.256):0.864):0.011);
Tree gt32=(d:2.582,(a:2.139,((b:1.062,c:1.062):0.667,(e:1.105,f:1.105):0.624):0.41):0.443);
Tree gt33=((b:1.112,c:1.112):2.227,(a:2.025,(d:1.491,(e:1.163,f:1.163):0.328):0.534):1.315);
Tree gt34=((f:1.411,(d:0.744,e:0.744):0.667):1.881,(a:1.84,(b:1.241,c:1.241):0.599):1.452);
Tree gt35=(a:2.415,((b:1.132,d:1.132):1.098,(e:1.571,(c:1.517,f:1.517):0.055):0.659):0.185);
Tree gt36=(a:2.426,((c:1.506,(d:0.846,e:0.846):0.66):0.281,(b:1.677,f:1.677):0.111):0.638);
Tree gt37=((c:1.617,(f:1.018,(d:0.565,e:0.565):0.453):0.598):0.624,(a:1.735,b:1.735):0.506);
Tree gt38=(a:3.036,((e:1.187,f:1.187):1.057,(c:1.497,(b:1.263,d:1.263):0.234):0.747):0.793);
Tree gt39=(a:2.492,((b:1.262,(c:0.807,d:0.807):0.455):0.725,(e:1.412,f:1.412):0.574):0.505);
Tree gt40=(a:3.378,((e:1.249,(d:1.116,f:1.116):0.132):0.311,(b:1.344,c:1.344):0.215):1.819);
Tree gt41=(a:2.082,(c:1.812,((d:0.542,e:0.542):1.268,(b:1.571,f:1.571):0.239):0.0020):0.27);
Tree gt42=((e:1.845,f:1.845):0.27,(a:1.965,(b:1.187,(c:0.715,d:0.715):0.471):0.778):0.15);
Tree gt43=(a:2.845,((f:1.296,(d:0.605,e:0.605):0.691):0.442,(b:1.501,c:1.501):0.237):1.107);
Tree gt44=(a:2.504,(b:2.482,(e:1.809,(f:1.565,(c:1.129,d:1.129):0.436):0.244):0.673):0.021);
Tree gt45=(a:2.758,((e:1.214,(d:1.008,f:1.008):0.206):0.433,(b:1.499,c:1.499):0.148):1.111);
Tree gt46=((f:1.165,(d:0.708,e:0.708):0.457):1.122,(a:1.873,(b:1.466,c:1.466):0.406):0.415);
Tree gt47=(a:2.406,((d:1.154,(e:1.117,f:1.117):0.037):1.104,(b:1.612,c:1.612):0.646):0.148);
Tree gt48=(a:2.155,(f:1.603,(e:1.55,(b:1.228,(c:0.504,d:0.504):0.724):0.322):0.053):0.552);
Tree gt49=(a:2.831,((e:1.492,f:1.492):0.176,(d:1.53,(b:1.401,c:1.401):0.13):0.138):1.163);

END;


BEGIN PHYLONET;

InferNetwork_MPL (all) 2;

END;
```

Command References

- Y. Yu and L. Nakhleh. A maximum pseudo-likelihood approach for phylogenetic networks. Under review.
- Z. Cao, J. Zhu and L. Nakhleh. Empirical performance of tree-based inference of phylogenetic networks.

See Also

- List of PhyloNet Commands