

GARLI 0.96 beta settings cheat sheet

(CDC Workshop, Feb 08)

This is an extremely abbreviated description of the settings in the `garli.conf` file. See the soon to be completed version 0.96 manual for much more detail. The old version 0.95 manual may be helpful for many settings.

See the end of this document for some practical guidelines on search strategies and appropriate program settings.

Conventions: allowable values or ranges of values for each setting appear in ()'s, default values in **bold**.

[general] section

datafname = <filename>

File containing the alignment, in Nexus, Phylip or Fasta format

ofprefix = <output prefix>

Prefix used for output files

streefname = (random, **stepwise**, <filename>)

Start run with random tree, random stepwise-addition tree or tree from file

attachmentspertaxon = (any #, **50**)

When streefname = stepwise, the number of attachments considered per taxon

constraintfile = (**none**, <filename>)

File containing constraint tree definition

searchreps = (any #)

The number of independent searches to perform

outputeachbettertopology = (**0**,1)

Whether to write improved trees encountered during the heuristic search to the file <ofprefix>.treelog00.tre. Not very useful except for learning purposes. Can create VERY large files during long runs.

outputcurrentbesttree = (**0**,1)

Whether to write best tree found up to that point to file <ofprefix>.current.best.tre every saveevery generations

Settings affecting automatic termination condition:

enforcetermconditions = (**0**,1)

Whether to use automatic termination conditions. Should be left at 1.

genthreshfortopterm = (1000 – 100,000, **10,000**)

Once precision parameter has reached minimum value, the number of generations without a topology improvement before the run is automatically terminated.

scorethreshforterm = (0.001 – 1.0, **0.05**)

Once precision parameter has reached minimum value, the threshold amount of improvement in log-likelihood over recent generations allowed for termination
significanttopchange = (0.001 – 1.0, **0.01**)
The minimum improvement in score needed for a topology to be considered “new” for purposes of the termination condition

Checkpointing:

writecheckpoints = (0,1)

Whether to write checkpoint files that can be used to restart to the program if it must be terminated (system crash, reboot needed, etc)

restart = (0,1)

Whether to restart a run from previously written checkpoint files

Model settings (see the manual for more detail on the allowable models and references):

datatype = (dna, aminoacid, codon-aminoacid, codon)

Type of model to use for the analysis. Protein coding DNA sequences aligned in frame can be analyzed at the amino acid level without having to manually translate them by specifying the codon-aminoacid datatype (however, note that currently only the standard genetic code is implemented)

ratematrix

for DNA or Codons (1rate, 2rate, 6rate, fixed) – The relative nucleotide substitution matrix. 1, 2 or 6 rate is the number of relative rates to estimate

for Amino acids (poisson, dayhoff, jones, wag) – The fixed amino acid matrix to use. Poisson means all amino acid substitutions are equivalent.

statefrequencies – the equilibrium state frequencies to use

for DNA (equal, empirical, estimate)

for Amino acids (equal, empirical, dayhoff, jones, wag)

for Codons (equal, empirical, F1x4, F3x4)

ratehetmodel

for DNA or Amino acids (none, gamma, gammafixed)

for Codons (none, nonsynonymous)

numratecats = (1 or greater, default is 4)

for ratehetmodel none this must be 1

for ratehetmodel gamma, the number of discrete rates to assume

for ratehetmodel nonsynonymous, the number of discrete nonsynonymous/synonymous rate ratios (aka omegas)

invariantsites = (none, estimate, fixed)

whether to allow and estimate a class of sites with a rate of zero (cannot change) not allowed for codon data

Miscellaneous settings:

randseed = (any #)

Random number seed to use, or -1 to determine from system time

availablememory = (#, **512**)

The approximate maximum amount of memory that GARLI should use, in megabytes

logevery = (any #, **10**)

Frequency (in generations) to write current time and log-likelihood to file <ofprefix>.log00.log

saveevery = (any #, **100**)

Frequency (in generations) to write current best tree to file or to write checkpoint files

refinestart = (0,1)

Whether to perform optimization phase before start of actual search. Should be left at 1

outputphylip = (0,1)

Whether to output the best tree found by the program in the phylip format in addition to the default Nexus format

outputmostlyuselessfiles = (0,1)

Self explanatory. You probably have no use for the files that would be output if this is 1

[master] section (you probably don't need to make changes here except in the case of bootstrapping)

bootstrapreps = #

Number of bootstrap replicates to perform. Note that if **searchreps** setting is > 1 then that number of searches will be done per bootstrap replicate, and the single best tree found across all of the search replicates will be written to the .boot.tre file.

Genetic algorithm settings:

nindivs = 4

Number of individuals the in the genetic algorithm population

holdover = 1

Number of copies of the best individual in the genetic algorithm population that are copied unmodified to the next generation.

selectionintensity = (0.01 – 1.0, **0.5**)

Strength of selection in genetic algorithm population. Larger numbers mean stronger selection.

stopgen = (large #)

Maximum number of generations to run. Generally should be set very large to allow the automatic termination condition to be used.

stoptime = (large #)

Maximum number of seconds to run. Generally should be set very large to allow the automatic termination condition to be used.

startoptprec = (0.001 – 1.0, default **0.5**)

The branch-length optimization precision at the start of a search. Lower values mean more intensive optimization.

minoptprec = (0.001 – 1.0, default **0.01**)

The branch-length optimization precision at the end of a search. Lower values mean more intensive optimization.

numberofprecreductions = (0 – 100, default **10**)

The number of times to decrease the branch-length optimization precision over the course of the run

treerejectionthreshold = (0.0 – 1000.0, default **50.0**)

The difference in lnL score between the current best tree and a newly proposed tree above which the new tree is rejected without full optimization

topweight = (0 – 10, **1.0**)

The relative weight assigned to all topology mutations. Setting to zero effectively fixes the topology and only estimates model parameters and branch lengths

modweight = (0 – 10, **0.05**)

The relative weight assigned to all model parameter mutations

brlenweight = (0 – 10, **0.2**)

The relative weight assigned to branch-length parameter mutations

randnniweight = (0 – 10, **0.1**)

Among topological mutations, the relative weight assigned to NNI (very local) topology mutations

randsprweight = (0 – 10, **0.3**)

Among topological mutations, the relative weight assigned to SPR (the most radical) topology mutations

limsprweight = (0 – 10, **0.6**)

Among topological mutations, the relative weight assigned to limited SPR (somewhat local) topology mutations

intervallength = (20 – 1000, **100**)

Length (in generations) of each “update interval”, the period after which the performance of the various mutation types is evaluated and adjusted

intervalstore = (1 – 50, **5**)

The number of “update intervals” to store and use to adjust mutation probabilities

inferinternalstates = (0,1)

Whether to infer (reconstruct) the marginal ancestral sequences at internal nodes. Currently only implemented for dna models.

How should I use the program, and what settings affect runtimes?

The primary settings that will affect runtimes are **genthreshfortopoterm**, **scorethreshforterm** and **numberofprecreductions**. There is always a tradeoff between the thoroughness of each search replicate (and how likely it is to find the global optimum) and the number of replicates that can be completed in a given amount of time. Unless the dataset is fairly large (greater than 200 sequences or so), it often seems to be better to perform many search replicates from different starting points than to make each search very intensive.

To start with, I suggest performing at least 5 search replicates from random or stepwise starting trees. If the best scoring tree is found multiple times among those searches, you can feel fairly confident that you've found a good tree (and hopefully the ML tree). For many datasets, even fast searches from random trees consistently result in the same tree. If there is significant variability among the replicates, try bumping up the search thoroughness and do more replicates. If you are still seeing variation, your best option is to keep doing search replicates until you stop finding new better scoring trees, and take the best of those as your ML tree.

For the purposes of bootstrapping, you can probably get away with somewhat less thorough searches for each replicate.

Very general guidelines for search settings:

Fast search:

streefname = stepwise
attachmentspertaxon = 5
gentreshfortopoterm = 5000
numberofprereductions = 1
treerejectionthreshold = 20

Moderately intense search (the default values):

streefname = stepwise
attachmentspertaxon = 50
gentreshfortopoterm = 10000
numberofprereductions = 10
treerejectionthreshold = 50

Intensive search (this might take a long time on large datasets):

streefname = stepwise
attachmentspertaxon = 50
gentreshfortopoterm = 20,000 to 100,000
numberofprereductions = 20 to 40
treerejectionthreshold = 100