A Likelihood Framework to Measure Horizontal Gene Transfer (HGT)

Simone Linz, Achim Radtke, Arndt von Haeseler
If you have further questions, please send an email to
arndt.von.haeseler@univie.ac.at or linz@cs.uni-duesseldorf.de!

Content of the directory

A more user-friendly version will be available soon!

hgt_simulate

Programs to simulate HGT events and further library stringtree.cpp.

- hgt_sim.cpp:
  Simulates m times n HGT events on the species tree given in the species tree file. Species tree must be in string format and the HGT rate is homogeneous.
  Usage: hgt_sim n m species_tree_file

- hgt_simrate.cpp:
  Simulates m times a HGT rate λ on the species tree given in the species tree file. Species tree must be in string format and the HGT rate is homogeneous.
  Usage: hgt_simrate λ m species_tree_file

quartet_finder

Program calculates how often each quartet tree (randomly chosen over the set of all gene tree quartet trees) exists in the simulated trees (sim_file).
Usage: quartet_finder sim_file quartet_file #HGT_events #species

subtree_finder

Program calculates how often each gene tree exists in the simulated trees (sim_file). (similar to the program quartet_finder)
Usage: subtree_finder sim_file quartet_file #HGT_events #species

count_subtree_matches

Program calculates the sum over all columns of the output file of the program subtree_finder or quartet_finder.
Usage: count_subtree_matches infile #HGT_events #quartets

matrix2lh_val

Program reads in a matrix file and performs the maximum likelihood estimation procedure, starting at start_rate. The tree_length is the total branch length of the species tree and the #simulations is how often you have simulated for each number of HGT events.
Usage: matrix2lh_val matrix_file start_rate stop_rate step_rate tree_length #simulations

baeume, compares, and filefctns

C libraries, scripts for compilation are inside each directory

newick2string

Generates the string format of a tree in newick format (without branch lengths).
Usage: newick2string infile

newick2string_with_branch_lengths

Generates the string format of a tree in newick format (with branch lengths). Note that each branch length represents the current branch lengths and not

the sum over all branch lengths form the root to the corresponding node.
Usage: newick2string_with_branch_lengths infile

string2split

Program converts the string notation of a set of gene trees into the split notation. To do so, the Perl module TreeFormat.pm is necessary which is also included in this directory.
Usage: perl string2split.pl gene_trees_infile #gene_trees

string2newick

Generates the newick format of the infile tree in string format (without branch lengths). Note that the infile must contain an empty line at the end. Usage: string2newick infile

further_scripts

collection of other short scripts which might be helpful:

- cat_sort.sh:
  Generates a matrix in which each row i represents a number of HGT events ($1..n$) and each column j a gene tree (e.g. a quartet tree). Then, the value v_{ij} tells you how often the quartet j occurs in the m simulated trees, each with i HGT events.

- compare_topologies.pl:
  Calculates how often the species tree topology exists in the simulation file containing a set of tree topologies. All trees must be in string format.
  Usage: perl compare_topologies.pl species_tree_file simulations_file

- create_random_quarts.pl:
  Creates n random quartet trees from a set of species.
  Usage: perl create_random_quarts.pl n

- extract_all_quartets.pl:
  Extracts quartet subtrees of a given gene tree file.
  Usage: perl extract_all_quartets.pl gene_tree_file
- extract_random_quarts.pl:
  Draws n quartet trees from a file containing a list with quartet trees in splits notation without replacements in a random manner.
  Usage: perl extract_random_quarts.pl n
- make_final_matrix.sh:
  Generates final matrix as input for the maximum likelihood estimation. Therefore, the very first line of the matrix (for zero HGT events) must be calculated. To start the script you have to edit the first line.
- submit.sh and count.sh:
  See workflow below!

examples

some test files

ranlib

Library of C routines for random number generation.
(source: http://orion.math.iastate.edu/burkardt/c_src/ranlib/ranlib.html)

Typical Workflow

Assuming that a clocklike rooted species tree with corresponding branch lengths in the string format (examples/cog_species_tree.string) and a set of gene tree topologies in the split format (examples/cog_gene_trees.split) are given, a typical workflow could look like the following:

1. Simulate m times n HGT events on the given species tree, usually with $m = 100,000$ and $1 \leq n \leq 180$.
   (one of the programs in the hgt_simulate directory)

2. Extract all quartet subtrees of the given gene tree collection.
   (extract_all_quartets.pl)

3. Generate several sets of randomly chosen quartet subtrees, usually 10,000 subtrees per set.
   (extract_random_quarts.pl)

4. Generate a matrix in which each row i represents a number of HGT events ($1..n$) and each column j a gene tree (e.g. a quartet tree). Then, the value v_{ij} tells you how often the quartet j occurs in the m simulated trees, each with i HGT events.
   (quartet_finder, count_subtree_matches, cat_sort.sh, make_final_matrix.sh)

5. Estimate the HGT rate λ using a maximum likelihood estimation procedure.
   (matrix2lh_val)
   $\#HGTs = \lambda \cdot L$, with L equal total branch length of the species tree.

Some remarks:

- Programs for which no usage is given above can be started by just typing in the program name.
- The species tree must be clocklike.
- The tree (((a:1,b:1):1,c:2):2,d:4); has the corresponding string format abcd abc ab a b c d. If branch lengths are required the following is a valid format abcd:0 abc:2 ab:3 a:4 b:4 c:4 d:4. This means that each branch length is the sum over all branch lengths from the root to the corresponding node. For clocklike trees, each leaf must have the same distance to the root!