SimPhy 1.0 Manual

1. About SimPhy

SimPhy is a program for the simulation of gene family evolution under incomplete lineage sorting (ILS), gene duplication and loss (GDL), replacing horizontal gene transfer (HGT) and gene conversion (GC). *SimPhy* simulates species, locus and gene trees with different levels of rate heterogeneity, and uses INDELible (Fletcher and Yang, 2009) to evolve nucleotide/codon/aminoacid sequences along the gene trees. The input for *SimPhy* are the simulation parameter values, which can be fixed or sampled from user-defined statistical distributions. The output consists of sequence alignments and a relational database that facilitate posterior analyses.

2. Citation

Please, if you use *SimPhy*, cite:

Mallo D, de Oliveira Martins L, Posada D (2015) **SimPhy: Phylogenomic Simulation of Gene, Locus and Species Trees**. Syst. Biol. doi: http://dx.doi.org/10.1093/sysbio/syv082

3. Getting help

Most common issues, doubts and questions should be solved reading this manual. If it is not the case or you find any bug, you can post your question at *SimPhy*'s google group: https://groups.google.com/forum/#!forum/simphy .

4. Obtaining SimPhy

SimPhy has been designed to work and tested in Linux and Mac OSX. If you are interested in using it in a Windows SO, let us know and we will try to port it.

4.1 Download

SimPhy's source code, together with precompiled versions, is available from *SimPhy*'s GitHub repository: https://github.com/adamallo/SimPhy. You can get the latest release with all accompanying data from https://github.com/adamallo/SimPhy/releases/latest or clone only the sources in your computer executing "git clone https://github.com/adamallo/SimPhy".

4.2 Precompiled versions: The easiest way

Ideal for users who just want to use the software. You only need to download the appropriate binary file for your system, and put it in the folder you would like to run it from.

- Linux executable 32-bits: simphy_lnx32 and simphy_lnx32_lsb
- Linux executable 64-bits: simphy_lnx64 and simphy_lnx64_lsb
- MacOSX executable 64-bits: simphy_mac64

You may need to set the execution permissions "chmod +x simphy_XXXXX" in order to use them. Linux executables with the suffix "_lsb" compliant with the Linux Standard Base,

while those without the suffix have been compiled with the GNU Compiler Collection (GCC).

4.2b Compilation and installation

If you are not interested in running SimPhy in an unsupported OS, reading or modifying the sources, we recommend you against compiling it by yourself, and refer you back to 4.2. For the rest, this section describes how to compile SimPhy from source.

SimPhy depends on four libraries:

- GNU Scientific Library (GSL): http://www.gnu.org/software/gsl/
- GNU Multiple Precision Arithmetic Library (GMP): https://gmplib.org/
- GNU MPFR Library http://www.mpfr.org/
- SQLite3 http://www.sqlite.org/

You can find easy installation guidelines in their own documentation. Once the dependences are installed, *SimPhy* can be easily compiled just running "make" in the main folder (SimPhy). The generated binary file will be placed in bin/simphy.

If you do not have admin privileges you may need to install the libraries in alternative directories and then indicate them at both compilation and execution times. Below you can find a comprehensive example doing so for users with the four dependences not fulfilled.

```
#Step 0: Downloading and extracting the sources

#Version numbers and links may change with the time, this as an example and adapt it to your needs

mkdir -p /home/user/temp

cd /home/user/temp

wget http://www.sqlite.org/2015/sqlite-autoconf-3080900.tar.gz #SQlite3 sources

wget ftp://ftp.gnu.org/gnu/gsl/gsl-latest.tar.gz #GSL sources

wget https://gmplib.org/download/gmp/gmp-6.0.0a.tar.bz2 #GMP sources

wget http://www.mpfr.org/mpfr-current/mpfr-3.1.2.tar.gz #MPFR sources

git clone https://github.com/adamallo/SimPhy #SimPhy sources

tar -xvzf gsl*

tar -xvzf gmp*

tar -xvzf mpfr*

tar -xvzf sqlite*

mv gsl* gsl
```

```
mv gmp* gmp

mv mpfr* mpfr

mv sqlite* sqlite3

#Step1: SQLite3 compilation and installation

cd /home/user/temp/sqlite3

./configure --prefix=/home/user/opt/local

make

make install

make clean

#Step2: GSL compilation and installation

cd /home/user/temp/gsl

./configure --prefix=/home/user/opt/local

make

make install

make clean

#Step3: GMP compilation and installation

cd /home/user/temp/gmp

./configure --prefix=/home/user/opt/local

make

make install

make clean

#Step4: Environment configuration

export C_INCLUDE_PATH=/home/user/opt/local/include

export LD_LIBRARY_PATH=/home/user/opt/local/lib

export LDFLAGS=-L/home/user/opt/local/lib

echo       'export       LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/home/user/opt/local/lib'       >>
/home/user/.bashrc

#Step5: MPFR compilation and installation
```

```
cd /home/user/temp/mpfr

./configure --prefix=/home/user/opt/local --with-gmp=/home/user/opt/local

make

make install

make clean

#Step6: SimPhy compilation

cd /home/user/temp/simphy

make

make clean

#You have obtained the simphy binary executable, placed in /home/user/temp/simphy/bin. Put it wherever you want to use it from.
```

5. Usage

SimPhy obtains its arguments from the command-line and/or from a configuration file. A simple script (scripts/INDELible_wrapper.pl) is provided in order to simulate multiple sequence alignments with INDELIble along the resulting gene trees.

A typical simulation scenario would start simulating a species tree, followed by the simulation of a number of locus trees and the subsequent simulation of a gene tree per locus tree. Nevertheless, this scheme can be deeply modified. For example, it is possible to start from a fixed species tree provided by the user, or directly from a collection of locus trees. For each replicate most simulation parameters can be sampled from flexible statistical distributions.

5.1 Configuration

SimPhy parses its arguments from the command line and/or from a configuration file. The arguments follow the standard notation for command-line options (a hyphen followed by a tag to identify the option, followed by a value, e.g, -a 0.3). Both command-line and a configuration file arguments can be specified simultaneously, with command-line arguments having higher priority in case of conflict.

5.1.1 Input parameters

Parameter types are indicated using the following nomenclature: i = integer, r = real, c = character string, b = boolean (1,0), * = *sampling notation* (see 5.1.2).

- -R... → *Replicates*
  - -RS i: Number of species tree replicates (i.e., study replicates).
  - -RL *: Number of locus trees per species tree.
  - -RG i: Number of gene trees per locus tree. ***(Not for general usage).***

- -G... → *Genome-wide parameters* (sampled for each species tree)
  - -GB *: Duplication parameter (to use with LB).
  - -GD *: Loss rate parameter (to use with LD).
  - -GT *: Transfer parameter (to use with LT).
  - -GG *: Gene conversion parameter (to use with LG).
  - -GP *: Gene-by-lineage-specific parameter (to use with HG) .
- -S... → *Species tree parameters*
  - -S c: Species tree (extended Newick format) (see 5.2.1).
  - -SR c: Nexus file with species trees (see 5.2.2).
  - -SB *: Speciation rate (events/time unit).
  - -SD *: Extinction rate (events/time unit).
  - -ST *: Species tree height (time units).
  - -SL *: Number of taxa .
  - -SO *: Ratio between ingroup height and the branch from the root to the ingroup. If this parameter is not set the outgroup is not simulated.
  - -SI *: Number of individuals per species.
  - -SP *: Tree-wide effective population size.
  - -SU *: Tree-wide substitution rate.
  - -SG *: Tree-wide generation time.
- -L... → *Locus tree parameters*
  - -L c: Locus tree (modified Newick format) (see 5.2.1).
  - -LR c: Nexus file containing locus trees (see 5.2.2).
  - -LB *: Duplication rate (events/generation).
  - -LD *: Loss rate (events/generation).
  - -LT *: Horizontal gene transfer rate (HGT) (events/generation).
  - -LG *: Gene conversion rate (GC) (events/generation).
  - -LK b: Distance-dependent HGT/GC: Determines if the sampling of receptors of genetic material depends (1) or not (0) on the evolutionary distance between candidates and donors.
  - -LL i: Minimum number of locus tree leaves.
  - -LS i: Minimum number of species represented by the locus tree leaves.
- -H...→ *Substitution rate heterogeneity parameters*
  - -HS *: Species-specific branch rate heterogeneity modifiers.
  - -HL *: Gene-family-specific rate heterogeneity modifiers.
  - -HH *: Gene-by-lineage-specific locus tree parameter (to use with the HG argument below)
  - -HG *: Gene-by-lineage-specific rate heterogeneity modifiers.
- -C... → *Global options*
  - -CS r: Random number generator seed.
  - -CE r: Precision of the Brent's method for root-finding when sampling the multilocus coalescent. ***(Not for general usage)***
- -I c: Input configuration file.
- -V [0,6]: Verbosity **(Note: the larger the verbosity the slower *SimPhy* becomes. Levels over 3 may only make sense for debugging or study how *SimPhy* works.)**
  - -V 0: Only warnings and errors.
  - -V 1: Global settings summary, simulation progress per replicate (number of simulated gene trees), warnings and errors.

- -V 2: Global settings summary, simulation progress per gene tree, warnings and errors.
- -V 3: Global settings summary, sampled settings per species, locus and gene trees, simulation progress per gene tree, warnings and errors.
- -V 4: Global settings summary, sampled settings per species, locus and gene trees, simulation progress per gene tree, simulated trees, warnings and errors.
- -V 5: Global settings summary, sampled settings per species, locus and gene trees, simulation progress detailing the internal algorithms (events), simulated trees, IO progress, warnings and errors.
- -V 6: Global settings summary, sampled settings per species, locus and gene trees, simulation progress detailing the internal algorithms with the highest detail (event times, rejections, probabilities…), simulated trees, IO progress, warnings and errors.

- -O...→ *Output parameters*
  - -O c: Common output prefix-name (for folders and files).
  - -OT b: Determines whether the species and locus tree branches are written in number of generations (0) or time units (1).
  - -OM b: Activates the tree mapping output.
  - -OD b: Activates the SQLite database output.
  - -OP b: Activates logging of sampled options.
  - -OC b: Activates the logging of original command line parameters and input configuration files.
  - -OL b: Activates the output of trees with internal nodes labeled by its post-order id starting from 0 (see 5.3).
  - -ON b: Activates the output of the bounded locus subtrees file (see 5.3).

5.1.2 Sampling notation

Certain *SimPhy* parameters have to be given using a specific notation in order to unambiguously define statistical distributions and dependence between arguments. This allows us to sample them for each replicate, either at the species tree (GB,GD,GT,GG,GP,RL,SB,SD,ST,SL,SO,SI,SP,SU,SG,HS,HL), locus tree (LB,LD,LT,LG,HH) or gene tree (HG) level. A statistical distribution code (see table below) is followed by a colon and a list of comma-separated parameter values in this notation.

Example:

```
N:5,0.5
```

Normal distribution with mean=5 and standard deviation=0.5

Distribution	**Code**	**Parameter 1**	**Parameter 2**	**Parameter 3**
Point (fixed value)	F	Value		

Uniform	U	Minimum (included)	Maximum (excluded)	
Normal	N	Mean	Scale	
Exponential	E	Rate		
Gamma	G	Shape	Scale	
LogNormal	LN	Location	Scale	
LogUniform	LU	Minimum	Maximum	
Lognormal * constant	SL	Location	Scale	Multiplier

These parameters can be interdependent (as far as the dependences do not generate loops), and are specified with their name.

5.1.3 Dependence between parameters

Dependence between parameters has been implemented in order to parameterize simulations in a hierarchical way. Thus, parameters sampled in the upper simulation layers can act as hyper-parameters for lower layers, modulating any characteristic (mean, variance…) of the considered evolutionary process. This simulation strategy makes the downstream analysis of the data easier, since each study replicate (species tree) can have associated the expected intensity (for example) of the considered evolutionary process (e.g., replicates with low expected number of duplications vs replicates with high expected number of duplications). **Locus-tree simulation** (i.e., duplication, loss, transfer and gene conversion rates) and **substitution rate heterogeneity** relay by default in this characteristic.

In the first case, genome-wide parameters (GB,GD,GT and GG) are sampled for each species tree from user-specified distributions, which model the variability of the evolutionary processes (i.e., duplication, loss, transfer, gene conversion) in our considered biological systems. These genome-wide values are subsequently used to parameterize the distributions to sample one rate per locus tree (LB,LD,LT and LG). This scheme allows some study replicates to have a much higher or lower expected number of events, but still variability among locus trees. The linkage between the hyper-parameters (GB,GD,GT and GG) and parameters (LB,LD,LT and LG) is determined by the user, and therefore the first can modify different characteristics of the process (mean, variance…).

Example:

```
-gb u:-25,-15 -lb l:gb,0.4 -gt l:-20,0.4 -lt f:gt -gg u:0.5,1 -lg l:-30,gg
```

Duplication rate: The genome-wide parameter is sampled from an U(-25,-15) for each species tree. Then, this value is used as the meanlog to sample the birth rate from a lognormal with sdlog=0.4. Thus, the genome-wide parameter mainly determines the intensity of the duplication rate. **Transfer rate:** The genome-wide parameter is sampled from a Lognormal(-20,0.4) for each species tree, and

this value is directly used for all locus trees. Therefore, the genome-wide parameter is determining the transfer rate, and variability among loci is not being considered. **Gene conversion rate:** The genome-wide parameter is sampled from a U(0.5,1) for each species tree, and then used as sdlog for a Lognormal with meanlog=-30. Thus, in this example the genome-wide parameter mainly controls the variance among locus trees of the transfer rate for each replicate.

Substitution rate heterogeneity follows a similar strategy (see 6 for more details on the model). Thus, *SimPhy* incorporates genome-wide parameters for the three different kinds of heterogeneity (HS,HL,GP). For the species-specific and gene-family-specific heterogeneities these values (sampled per species tree) are used as the alpha parameter of a Gamma distribution with mean one, which is sampled to obtain multipliers per species tree branch (species-specific) or the whole locus tree (gene-family-specific) respectively. For the third kind of heterogeneity (gene-by-lineage-specific), there are two extra layers of complexity. Thus, the genome-wide value is used to parameterize a distribution (HH parameter), which is sampled per locus tree, and in turn is used to parameterize another distribution (HG parameter) sampled per each gene tree, which is finally used as alpha parameter to sample multipliers for each gene tree branch from a gamma distribution with mean one.

Example:

```
-hs l:1.5,1 -hl l:1.2,1 -gh u:1,2 -hh l:gh,1 -hg f:hh
```

Species-specific and **gene-family-specific** genome-wide parameters are sampled from a Lognormal(1.5,1) and Lognormal(1.2,1) respectively, and then used as alpha parameter of a Gamma distribution to sample the substitution rate multipliers. The genome-wide parameter for the **gene-by-lineage-specific** heterogeneity is sampled from a U(1,2) for each species tree. Afterwards, this value parametrizes the meanlog of Log-normal with sdlog=1, sampled per locus tree. This value is used as the alpha parameter for the Gamma distribution sampled to get the multipliers for each gene tree branch. Therefore the variability among gene trees coming from the same locus tree is not considered.

Apart from these hierarchical dependences, any parameter can be linked to any other, as far as they can be sampled (i.e., there is no closed loops).

Example:

```
-ld f:lb
```

Loss rate linked to the duplication rate.

5.1.4 Configuration file format

The configuration file has been designed as a proxy for the command-line arguments. Therefore, it follows the same command-line-option-like format (see 5.1.1) but with each parameter placed in a different line. Moreover, text can be commented out just preceding it with "//". You can find an example configuration file (SimPhy_controlfile.conf) in the *SimPhy* distribution.

5.1.5 Environmental Variables

Some technical parameters, basically related to the computational implementation, can be set using environmental variables. These options should not be modified unless *SimPhy* returns an error suggesting/indicating to do so. Please, do not play with these variables if you do not really know what you are doing, since you may really slow *SimPhy* down or even make it crash.

SIMPHY_NUMBUFFER: Initial size in bytes of the input buffer for parsing floats. It is dynamically allocated and reallocated in order to avoid overflows, and therefore it is not necessary to modify this parameter.

SIMPHY_IOBUFFER: Initial size in bytes of the general input buffer. It is dynamically allocated and reallocated in order to avoid overflows, and therefore it is not necessary to modify this parameter.

SIMPHY_TESTCHAR: Character used to test for uncompleted buffer reads in certain Nexus trees input functions. The only condition is that it must not be present in the input files. By default BEL (ASCII 7) is used.

SIMPHY_MAXLEAVES: Fixed maximum number of locus tree leaves. This avoids memory/infinite-loop problems due to improper birth rates in either species or locus tree simulations. You may need to increase it for huge simulation studies (uncommon, please, do not do it a priori).

SIMPHY_MAXNAME: Fixed maximum length of a species name.

SIMPHY_MAXIT: Maximum number of iterations. This limit is used across in order to avoid infinite loops. You may need to increase it for very large simulation studies (uncommon, please do not modify it unless *SimPhy* suggests to do so).

SIMPHY_FLOATPREC: Maximum difference between two floats to be considered as equal. This value is used to perform the input tree ultrametricity test. You may need to lower it if you generated input trees with programs with low float precision.

5.2 Input tree files

If needed, starting trees (either species or locus trees) can be directly specified by the user. Note that certain **branch specific parameters** can be also defined (see below). When specifying the tree as a command-line parameter, a modified Newick tree format must be used (-S or -L). When specifying a tree-file (-SR or -LR) a Nexus tree format must be used.

5.2.1 Input files: Newick tree format

SimPhy uses an extended version of the original Newick tree format (see http://evolution.genetics.washington.edu/phylip/newicktree.html) in order to indicate **branch specific parameters**:

Code	Extended Newick species/locus tree branch specific parameters
:	Branch length (number of generations)

*	Substitution rate multiplier
~	Generation time multiplier
#	Effective population size
/	Number of individuals (**restricted to external branches)**
%	Node kind (0=Speciation, 1=Duplication, 2=Loss, 3=Transfer donor, 4=Transfer acceptor, 5=Gene conversion donor, 6=Gene conversion receptor) (**restricted to locus trees**)
_	Locus unique id (**restricted to locus trees**)

Example:

```
((A:20,B:20/2):15#300~2,C:50);
```

A→Branch length=20 generations; B→Branch length=20 generations, two individuals; AB→Branch length=15 generations, Effective population size=300, Generation time= twice the global; C→Branch length=50 generations. **Note that this tree is ultrametric in time units (see 5.2.3).**

5.2.2 Input files: Nexus tree format

The same branch specific parameters have been implemented in nexus trees as branch comments ([&param1=x,param2=y…]).

Code	Nexus species/locus tree branch specific parameters
:	Branch length (number of generations)
u_mult	Substitution rate multiplier
g_mult	Generation time multiplier
pop_size	Effective population size
n_ind	Number of individuals (**restricted to external branches)**
kind_n	Node kind (0=Speciation, 1=Duplication, 2=Loss, 3=Transfer donor, 4=Transfer acceptor, 5=Gene conversion donor, 6=Gene conversion receptor) (**restricted to locus trees**)
paralog	Locus unique id (**restricted to locus trees**)

Example (equivalent to the one in 5.2.1):

```
#NEXUS

begin trees;

	tree 1=((A:20,B:20[&n_ind=2]):15[&pop_size=300,g_mult=2],C:50);

end;
```

5.2.3 Species/locus input tree requirements

Branch length units for species and locus trees are always generations. Currently, we only consider ultrametric (in time) species and locus trees. Note that if the specified generation time is 1, generation units will be equivalent to time units.

5.3 Output files

SimPhy generates a main folder to store the simulation results, as indicated by the -O argument (X) (see 5.1). One subfolder per replicate is then created within this folder (identified by the replicate number), together with the database (X.db), original command line arguments (X.command, if required) a copy of the original input configuration file (X.conf, if used and required) and a file summarizing the sampled options (X.params). Inside the replicate subfolders the species tree is located in the "s_tree.trees" file, while the locus trees are saved in the "l_trees.trees". The "bounded_locus_subtrees.out" file specifies the bounded locus subtrees per locus tree and is generated if required (see 5.1 and 5.3.2). One gene tree output file is generated per locus tree (gene_treeY.trees, Y= locus tree id). Moreover, when required (see 5.1) mappings are also saved in this folder with different extension identifying the mapped trees (Y.mapsl, species tree / locus tree Y; YlZg.maplg locus tree Y/ gene tree Z).

5.3.1 Output tree format

Species and locus trees are written in Newick format, with branch lengths in number of generations (default) or time units (see -OT option in 5.1.1). Gene trees are also written in Newick format, with branch lengths in expected number of substitutions per site. Internal nodes are not identified with any tag by default, but can be tagged with their post-order id (see -OL option in 5.1.1).

5.3.2 Bounded locus subtrees output file format

The bounded_locus_subtrees.out file identifies the root nodes of the different subtrees modeled by a bounded multispecies coalescent present in each locus tree. Each line contains the comma-separated list of nodes (identified by their name if they are leaves or their post-order id if they are internal nodes, like the internal labels activated by the -OL option, see 5.1.1) present in one locus tree.

5.3.3 Species tree / locus tree mapping format

The species tree / locus tree mapping is returned as a table with one row per locus tree node (post-ordered) and 4 tabulated columns indicating the locus tree node, locus id, evolutionary event and corresponding species tree node. Nodes are identified using either its quoted name (external nodes) or a post-ordered id (starting from 0). The evolutionary

event indicates if an internal node is either a regular leaf (leaf) or a lost leaf, either by a regular loss (Loss) or reception of either a transfer (Rtransf) or gene conversion (Rgc). However, internal nodes can correspond to a speciation (Sp), duplication (Dup), transfer (Transf) or gene conversion (Gconv).

5.3.4 Locus tree / gene tree mapping format

The locus tree / gene tree mapping is also saved as a table with one row per gene tree node (post-ordered) and 3 tabulated columns indicating the gene tree node, locus id and corresponding locus tree node. Nodes are identified using either its quoted name (external nodes) or a post-ordered id (starting from 0).

5.3.5 Automatic leaf labeling format

Locus tree leaves are automatically labeled using modifications to the species name in order to indicate both the species and locus, following the scheme "species_locusid", while gene tree leaves also indicate the individual "species_locusid_individualid". Locus trees may also contain lost leaves. In this case the scheme indicates both the evolutionary event that generated them and the corresponding species tree node id "event-streenodeid". More than one lost leave can pertain to the same species tree node, and consequently some leaves can have the same name. When using the outgroup addition option (see 5.1.1) the outgroup species is called "0".

5.3.6 DB schema

SimPhy generates a SQLite database composed by 3 linked tables with different information about the species, locus and gene trees (respectively Species_Trees, Locus_Trees and Gene_Trees tables, see tables and schema below).

Species_Tree	
Field	**Description**
SID	Tree ID
Leaves	Number of leaves
SB_rate	Speciation rate
SD_rate	Extinction rate
F_length_gen	Fixed tree height (number of generations)
Height_cu	Tree height (coalescent units)
Length_cu	Tree length (coalescent units)
Outgroup_intlength_gen	Length of the branch linking the ingroup with the root, (number of generations)
Ind_per_sp	Number of individuals per species

N_loci	Number of loci
GB_rate	Genome-wide duplication parameter
GD_rate	Genome-wide loss parameter
GT_rate	Genome-wide transfer parameter
GGC_rate	Genome-wide gene conversion parameter
Alpha_s	Alpha parameter for the species-specific rate heterogeneity generator
Alpha_l	Alpha parameter for the gene-specific rate heterogeneity generator
Salpha_g	Alpha hyper-hyper-parameter for the gene-by-lineage-specific rate heterogeneity generator
Ne	Effective population size
Mu	Substitution rate
Gen_time	Generation time

Locus_Trees	
Field	**Description**
LID	Locus tree database ID
n_ltree	Locus tree replicate number for its containing species tree
SID	Containing species tree ID
b_rate	Duplication rate
d_rate	Loss rate
t_rate	Transfer rate
gc_rate	Gene conversion rate
n_leaves	Number of leaves
n_dup	Number of duplications
n_loss	Number of losses
n_transf	Number of transfers

n_gc	Number of gene conversions
gamma	Value to scale the gene trees evolved using this locus tree (gene-specific rate heterogeneity generator).
Lalpha_g	Alpha hyper-parameter for the gene-by-lineage-specific rate heterogeneity generator

Gene_Trees	
Field	**Description**
GID	Gene tree database ID
n_gtree	Gene tree replicate number for its locus tree
LID	Containing locus tree database ID
n_ltree	Containing locus tree replicate number for its containing species tree
SID	Containing species tree ID
Alpha_g	Alpha parameter for the gene-by-lineage-specific rate heterogeneity generator
n_leaves	Number of leaves
Extra_l	Number of extra lineages
Tree_h_cu	Tree height (coalescent units)
Tree_l_bl	Tree length (number of substitutions per site)

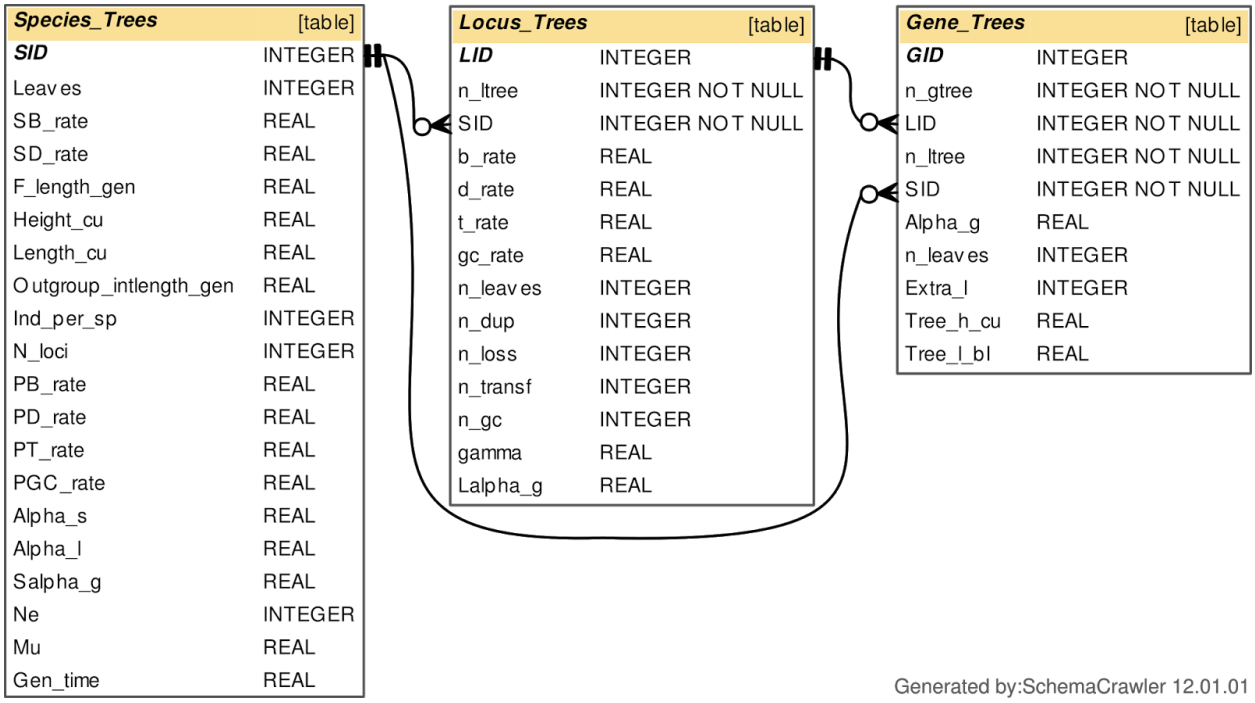

Fig. 1. Schema of the output relational database generated by SimPhy. ***Species_Trees table***: SID: species tree id, Leaves: number of species tree leaves, SB_rate: speciation rate, SD_rate: extinction rate, F_length_gen: fixed species tree length in number of generations, Height_cu: species tree height in coalescent units, Length_cu: species tree length in coalescent units, Outgroup_intlength_gen: length of the branch from the ingroup to the species tree root in number of generations, Ind_per_sp: number of individuals per species, N_loci: number of locus to simulate, PB_rate: genome-wide duplication parameter, PD_rate: genome-wide loss parameter, PT_rate: genome-wide horizontal gene transfer parameter, PGC_rate: genome-wide gene conversion parameter, Alpha_s: species-specific rate heterogeneity parameter, Alpha_l: gene-family-specific rate heterogeneity parameter, Salpha_g: gene-by-lineage genome-wide heterogeneity parameter, Ne: tree-wide effective population size, Mu: tree-wide substitution rate, Gen_time: tree-wide generation time. ***Locus_Trees table***: LID: locus tree id, n_ltree: gene family id relative to its simulation replicate, SID: corresponding species tree id, b_rate: duplication rate, d_rate: loss rate, t_rate: horizontal gene transfer rate, gc_rate: gene conversion rate, n_leaves: number of locus tree leaves, n_dup: number of duplication, n_loss: number of losses, n_transf: number of HGT events, n_gc: number of gene conversions, gamma: gene-family-specific multiplier, Lalpha_g: gene-by-lineage locus-related heterogeneity parameter. ***Gene_Trees table***: GID: gene tree id, n_gtree: gene tree id relative to its locus tree, LID: containing locus tree id, n_ltree: gene family id of the containing locus tree relative to its species tree, SID: corresponding species tree id, Alpha_g: gene-by-lineage heterogeneity parameter, n_leaves: number of gene tree leaves, Extra_l: number of extra lineages, Tree_h_cu: gene tree height in coalescent units, Tree_l_bl: gene tree length in expected number of substitutions per site.

5.4 Sequence simulation using INDELible_wrapper.pl

SimPhy includes a perl script to simulate sequence alignments using INDELible. If you use this script, please cite INDELible (Fletcher and Yang, 2009).

In order to use this script you will need a Perl interpreter, the Math::GSL Perl module and

an INDELible executable named "indelible" in your path. INDELible binaries can be downloaded for free from http://abacus.gene.ucl.ac.uk/software/indelible/download.php.

Set up example for Mac OS X:

```
cd Downloads
tar -xvzf INDELibleV1.03.gz
cp INDELibleV1.03/bin/indelible_1.03_OSX_intel ~/bin/indelible
#Make sure you have ~/bin in your PATH, or add it by doing, for example:
export PATH=$PATH:~/bin
#If you want to make it permanent, add it to your ~/.bash_profile and/or ~/.bashrc, for
example
echo "export PATH=$PATH:~/bin >> ~/.bashrc"
##Just in case your system does not have the Math::GSL Perl module installed run the
following line before using the INDELIible_wrapper.pl script
sudo cpan install Math::GSL
```

The INDELible_wrapper.pl uses a configuration file in which the user can specify different models for different loci (i.e., partitions). These simulations can be run in parallel if GNU Parallel (Tange, 2011) is available in your system. In order to increase the flexibility of this script, any numeric parameter can be specified using the same *sampling notation* explained above (see *5.1.2*), preceded by the dollar symbol and in brackets (e.g.,"$(U:10-20)"). Apart from the univariate distributions previously explained (see *5.1.2*), some multivariate distributions are also allowed:

Distribution	**Code**	**Parameters**
k-dimensional Dirichlet	D	$(\alpha_1, \alpha_2, ..., \alpha_k)$
k-dimensional Dirichlet with support relative to the latest element X($\frac{x_1}{x_k}, \frac{x_2}{x_k}, ..., \frac{x_{k-1}}{x_k}$)	RD	$(\alpha_1, \alpha_2, ..., \alpha_k)$

To properly set up the INDELible_wrapper configuration file users should refer first to INDELible's manual (http://abacus.gene.ucl.ac.uk/software/indelible/manual/). This file must include these blocks:

- [TYPE]: 1 block
- [SETTINGS]: optional block
- [MODEL] and/or [SIMPHY-UNLINKED-MODEL]: at least 1 block
- [SIMPHY-PARTITIONS]: at least 1 block
- [SIMPHY-EVOLVE]: 1 block

Including other type of blocks will result in an error message.

[TYPE], [SETTINGS] and [MODEL] blocks use the standard INDELible specification, but also accept parameters in *sampling notation* (see *5.1.2*).

A [SIMPHY-UNLINKED-MODEL] block is the same as a [MODEL] block except for the fact that parameter values inside this type of block will be sampled de novo for each gene family. Note that in the case of a [MODEL] block the parameter values specified will be the same for all gene families in a given replicate.

```
[SIMPHY-UNLINKED-MODEL] modelA

            [submodel] HKY $(e:1) // HKY with kappa sampled for every gene family with
parameter=1.

           [statefreq] $(d:1,1,1,1)  // frequencies for T C A G sampled from a Dirichlet
(1,1,1,1)
```

A [SIMPHY-PARTITIONS] block is used for each set of loci that will follow the same model (i.e., partition). Each block defines the name of the partition, the proportion of (consecutive) trees included in this partition and the sequence length at the root (fixed or sampled). For example:

```
[SIMPHY-PARTITIONS] simple [0.50 modelA $(n:1000,100)] // The first half of the gene
families will evolve under modelA. Their sequence lengths are sampled from a Normal with
mean=1000 and sd=100.

[SIMPHY-PARTITIONS] complex [0.50 modelB 500] // The second half of the gene families will
evolve under modelB. Their sequence lengths will be 500 bp in all cases.
```

A [SIMPHY-EVOLVE] block includes two options, the number of replicates, and the filename prefix. For example:

```
[SIMPHY-EVOLVE] 1 dataset // One sequence alignment for each gene tree, saved in files with
"dataset" as common prefix (it will generate dataset_1, dataset_2, etc.).
```

SimPhy includes two configuration file examples: "configuration_files/INDELible_control.txt" and "configuration_files/INDELible_complex.txt".

5.5 Short tutorial

With this short tutorial we want to show first-time users some SimPhy's basic functionalities in a glance. Moreover, in the next section (5.6) we provide more detailed examples. These configuration files can be used as templates to make it easier for the users to define their

own scenarios.

Case 1: ILS (1 replicate)

1) We will start executing the program help information. Thus, apart from getting a detailed explanation of the different simulation parameters, this will confirm that the installation has been successful.

```
simphy -h
   _____ _                 ____  __                    ___  ____
  / ___/(_)___ ___  ____  / __ \/ /_  __  __          <  / / __ \
  \__ \/ / __ `__ \/ __ \/ /_/ / __ \/ / / /          / / / / / /
 ___/ / / / / / / / /_/ / ____/ / / / /_/ /          / /_/ /_/ /
/____/_/_/ /_/ /_/ .___/_/ /_/ /_/\__, /           /_/(_)____/
                /_/                /____/

--------------------------------------------------


Getting settings from command line...
Usage: ./SimPhy -[Parameter code] value(i|r|c|b|*) ...

Value kinds
       i=integer
       r=real
       c=character string
       b=boolean(1 or 0)
       *=sampling notation

PARAMETERS
[...]
```

2) Next, we will simulate an scenario composed by 1 species tree, 1 locus tree and 1 gene tree, only considering ILS. To do so we will rely on the default settings, just providing the 3 parameters that are required in this case, a species birth rate, a number of species and an effective population size (0.000001 speciations per year, 10 species, 10000 individuals). As you can see in the command line below, we specify these parameters as fixed (f:).

```
simphy -sl f:10 -sb f:0.00001 -sp f:10000


   _____ _                 ____  __                    ___  ____
  / ___/(_)___ ___  ____  / __ \/ /_  __  __          <  / / __ \
  \__ \/ / __ `__ \/ __ \/ /_/ / __ \/ / / /          / / / / / /
 ___/ / / / / / / / /_/ / ____/ / / / /_/ /          / /_/ /_/ /
/____/_/_/ /_/ /_/ .___/_/ /_/ /_/\__, /           /_/(_)____/
                /_/                /____/

--------------------------------------------------

Getting settings from command line...Done
```

```
Global settings:
--------------------

Trees:
        -Species trees: 1 Birth-death simulations
                -Speciation rate: Fixed 1.000000e-05,
                -Extinction rate: Fixed 0.000000e+00,
                -Outgroup addition: No addition
                -Stopping rules:
                        -Number of leaves: Fixed 10,
                        -Generations: Not used
        -Locus trees: Fixed 1, directly obtained from each species tree (no birth-death
process)

        -Gene trees: 1 multilocus coalescent simulations

Parameters:
        -Haploid efective population size: Fixed 10000,
        -Generation time: Fixed 1.000000e+00,
        -Global substitution rate: Fixed 1.000000e+00,
        -Substitution rate heterogeneities
                -Lineage (species) specific rate heterogeneity gamma shape: No heterogeneity
                -Gene family (locus tree) specific rate heterogeneity gamma shape: No
heterogeneity
                -Gene tree branch specific rate heterogeneity gamma shape: Genome-wide
parameter (hyperhyperparameter) No heterogeneity, locus-tree related parameter No
heterogeneity, Gamma parameter(gene-tree related) No heterogeneity
        -Individuals per species: Fixed 1,

Misc parameters:
        -Rooting method epsilon: 0.000001
        -Seed: 4077170793112

I/O options:
        -Output files prefix: SimPhy_outfiles
        -Verbosity: 1
        -Stats file: OFF
        -Mapping: OFF
        -Database: OFF
        -Parameterization: OFF
        -Command-line arguments: ON
        -Bounded locus subtrees: OFF
        -Output trees with internal node labels: OFF


Simulation:
-----------
Replicate 1 of 1: Simulating 1 gene trees from 1 locus trees... Done
```

Before starting the simulation, *SimPhy* shows information on the parameterization of the simulation that it is going to carry out, and then it starts the process. Once the simulation has finished, you can find the simulated trees in the folder SimPhy_outfiles/1.

By default a lot of output options are deactivated. Nevertheless, for real applications it is advisable to activate most of them (only the mapping and daughters options can notably

increase the running time). However, increasing the verbosity level much further is not advisable, since it does not add much useful information for the regular user while it strongly affects the running time of *SimPhy*.

Case 2: HGT (100 replicates) + alignment simulation

Getting closer to a real simulation study, we will now imagine that we want to test the effect of HGT in a particular species tree.

1) We will start executing the program help as before. Thus, apart from getting a detailed explanation of the different simulation parameters, we will confirm that the installation has been successful.

2) Now we will indicate the species tree, population size (10000) and number of loci across replicates (100 genes). Instead of using a fixed HGT rate across all replicates, we will perform 100 replicates with HGT rates, constant across loci for a given replciate, and sampled from a distribution (Exponential with rate λ=10000). In order not to assume a strict molecular clock, we will apply a gene-by-lineage heterogeneity rate, with the same expected value for all gene trees. Afterwards, we will simulate a MSA 100 bp-long with INDELible, under an independent GTR+Γ for each loci, without indels.

Species tree:

```
(((A:10000,B:10000):30000,C:40000):1000,D:41000);
```

The species tree is specified in years. We will also specify that the generation time for these species is 0.5 years.

Here we have the configuration file:

```
-rs 100 //Number of replicates
-rl f:100 //100 locus per replicate
-s (((A:10000,B:10000):30000,C:40000):1000,D:41000); //Fixed species tree
-sg f:0.5 //Generation time
-sp f:10000 //Population size
-gt e:10000 //Genome-wide horizontal gene transfer (sampled once for each species tree, and
applied for all locus trees)
-hg f:100 //Heterogeneity sampled for every gene tree branch using the same Gamma
distribution with shape = rate =100
-cs 22 //Seed for the random number generator, in order to make the experiment repetible.
-om 1 //Tree mapping output
-od 1 //Database
-op 1 //Output with the general sampled options (describes the simulation run)
-oc 1 //Activates the backup of the original command line and configuration file (we
recommend to always activate this option)
```

3) Since we have not specified an output name in the input file, we will do it when we call the program (in a system with the SimPhy executable called simphy in the PATH variable

and the configuration file in the current directory named input_file_from_above.conf):

```
simphy -i input_file_from_above.conf -o tutorialHGT
```

4) After finishing the tree simulation, we will use this configuration file to run INDELible using the script INDELible_wrapper.pl, distributed with SimPhy:

```
[TYPE] NUCLEOTIDE 1     // DNA using algorithm 1

[SETTINGS]

        [fastaextension] fasta  // Fasta files with .fasta extension

[SIMPHY-UNLINKED-MODEL] simple_unlinked // Unlinked model (it will generate independent models for each tree using it, sampling parameter values for each of them)

     [submodel] GTR $(rd:6,16,2,8,20,4) // GTR with rates from a Dirichlet (6,16,2,8,20,4) scaled with the last rate (5 parameters to INDELible)

     [statefreq] $(d:1,1,1,1) // Equilibrium frequencies sampled from a Dirichlet (1,1,1,1)

     [rates] 0 $(e:2) 0 // Site-specific rate heterogeneities: 0 p-inv, alpha from an E(2) and using a continuous gamma distribution.)

[SIMPHY-PARTITIONS] simple [1 simple_unlinked $(f:100)] // All gene families will evolve under their own model originated by sampling the model simple_unlinked. Their sequence lengths are fixed to 100 nucleotides.

[SIMPHY-EVOLVE] 1 dataset // One sequence alignment for each gene tree, saved in files with "dataset" as common prefix (it will generate dataset_1, dataset_2, etc.).
```

Command (in a system with INDELible_wrapper.pl in the PATH variable and the configuration file called indelible_tuto.conf):

```
INDELIble_wrapper.pl tutorialHGT/ indelible_tuto.conf 22 1
```

Note that the last parameter is the number of cores you want to use in the simulation of sequences and using more than one requires the program GNU Parallel (http://www.gnu.org/software/parallel/). Even with one core, the simulation should finish in less than a couple of minutes.

For each replicate (folders 001-100) you can find one simulated multiple sequence alignment per gene (locus) in Phylip format (dataset_genenumber_TRUE.phy).

The experiment would further continue using a species tree reconstruction methods on those gene tree alignments and comparing the accuracies of different HGT levels (out of the scope of this tutorial).

5.6 Examples

-**Test case**: We will simulate 100 species trees with fixed height (1M generations) and variable number of taxa (U(20,49)), with fixed rates of duplication, loss and transfer (0.5 events per 1M generation), with U(10,99) locus trees per replicate and one gene tree per each. The effective population size (10000) and the substitution rate (0.00001) are fixed. Every possible output file will be written and placed in the directory "SimPhy_test".

```
simphy -sb f:0.000001 -ld f:0.0000005 -lb f:0.0000005 -lt f:0.0000005 -rs 100 -rl U:10,100
-rg 1 -o SimPhy_test -sp f:10000 -su f:0.00001 -sg f:1 -sl U:20,50 -st f:1000000 -om 1 -v 2
-od 1 -op 1 -oc 1 -on 1 -cs 22
```

-**INDELible_wrapper.pl for the Test case:** We will simulate one DNA alignment for every gene trees simulated. For each simulation replicate (i.e, for each species tree), 80% of the gene families will evolve under independent HKY models with kappa parameters coming from an Exponential(1) and equilibrium base frequencies from a Dirichlet (1,1,1,1). The remaining 20% gene families will evolve under a single GTR model (parameter values sampled once), with rates following a Dirichlet (6,16,2,8,20,4) scaled with the last rate (INDELible requirement), and equilibrium base frequencies sampled from a Dirichlet (1,1,1,1). Among-site rate-heterogeneity is modeled using a continuous gamma with shape sampled from an Exponential (2). Indels are simulated using a Zipfian distribution with parameter uniformly sampled from 1.5 to 2, and limited to gaps of size 10, with indel rate uniformly sampled from 0.001 to 0.002. Alignments for the first 80% gene families will have their length sampled from a Normal(1000,100), while the remaining ones will have their root sequence lengths sampled from a Normal(1000,10). One alignment will be generated per each gene tree and saved in files prefixed by "alignment". If GNU Parallel is available, two CPU cores will be used to simulate the sequences.

Execution code:

```
./scripts/INDELIble_wrapper.pl SimPhy_test configuration_files/INDELible_complex.txt 22 2
```

Configuration file:

```
[TYPE] NUCLEOTIDE 1     // DNA using algorithm 1

[SETTINGS]

        [fastaextension] fasta  // Fasta files with .fasta extension

[MODEL] complex_common //Shared by all the trees that use it (parameter values sampled only
once per study replicate)

                [submodel] GTR $(rd:6,16,2,8,20,4) // GTR with rates from a Dirichlet
```

```
(6,16,2,8,20,4) scaled with the last rate (5 parameters to INDELible)

            [statefreq] $(d:1,1,1,1) // Equilibrium frequencies sampled from a Dirichlet
(1,1,1,1)

          [rates] 0 $(e:2) 0 // Site-specific rate heterogeneities: 0 p-inv, alpha from an
E(2) and using a continuous gamma distribution.

         [indelmodel] POW $(u:1.5,2) 10 // Zipfian distribution with a parameter from 1.5 to
2 and a fixed maximum indel size of 10 nucleotides.

        [indelrate] $(u:0.001,0.002) // Insertion rate= deletion rate, ranging from 0.001 to
0.002 times the substitution rate.

[SIMPHY-UNLINKED-MODEL] simple_unlinked // Unlinked model (it will generate independent
models for each tree using it, sampling parameter values for each of them)

        [submodel] HKY $(e:1) // HKY with kappa sampled for every tree from an E(1)

           [statefreq] $(d:1,1,1,1)  // frequencies for T C A G sampled from a Dirichlet
(1,1,1,1)

[SIMPHY-PARTITIONS] simple [0.50 simple_unlinked $(n:1000,100)] // The first half of the
gene families will evolve under their own model originated by sampling the model
simple_unlinked. Their sequence lengths are sampled from a Normal with mean=1000 and sd=100.

[SIMPHY-PARTITIONS] complex [0.50 complex_common 500] // The second half of the gene
families will evolve under the model "complex_common". Their sequence lengths will be 500 bp
in all cases.

[SIMPHY-EVOLVE] 1 dataset // One sequence alignment for each gene tree, saved in files with
"dataset" as common prefix (it will generate dataset_1, dataset_2, etc.).
```

-**Complex parameterization example**: 100 replicates (i.e., 100 species trees) will be simulated. Genome-wide parameters will be sampled for each one: substitution rate (from an Exponential with rate=1E7), generation time (one generation per time unit), effective population size (from a Lognormal with meanlog=-14 and sdlog=1), duplication parameter (uniform -25 to -15), horizontal gene transfer parameter (linked to the duplication parameter), species specific heterogeneity parameter (Lognormal, meanlog=1.5, sdlog=1) and locus specific rate heterogeneity (Lognormal, meanlog=1.2, sdlog=1).

For the simulation of the species tree three parameters will be sampled, the speciation rate (from a Lognormal distribution with meanlog=-14, sdlog=1), the number of leaves (from an uniform 10-50) and the tree height (from an uniform 1E5-1E7 generations). An outgroup will diverge from the root of the tree by a distance equal to the ingroup tree height multiplied by a value sampled from a Lognormal distribution (meanlog=0, sdlog=0.1), the number of individuals per species will be uniformly sampled from 1 to 5 in every species, and substitution rate multipliers will be sampled for every branch from a Gamma distribution with both shape and rate given by the genome-wide species specific heterogeneity parameter.

The number of gene families or loci will be sampled from a uniform 10-500. The duplication rate will be obtained from a Lognormal distribution, with a meanlog given by the

genome-wide duplication parameter and sdlog=0.4. The horizontal gene transfer rate is parameterized in the same way as the duplication rate (meanlog=genome-wide transfer parameter, sdlog=0.4), but both will be kept as independent parameters. The loss rate is forced to be the same as the duplication rate (i.e., they are linked). The simulation of horizontal gene transfers is set to be distance dependent (i.e., it will more likely to happen between closely related lineages). The substitution rate associated to the branches of the locus tree will be scaled by a multiplier sampled from a Gamma with both shape and rate given by the genome-wide locus specific heterogeneity parameter.

Substitution rate multipliers will be sampled from a Lognormal (meanlog=1.4,sdlog=1) for every branch of the gene tree, and used to introduce among-lineage rate variation. In this example the expected gene-branch-specific substitution rate heterogeneity will be the same across species and gene families (otherwise we would need to introduce additional (hyper) parameters, as explained in 5.1.3). Several output files will be produced and placed in the directory "SimPhy_complex".

```
simphy -rs 100 -rl u:10,501 -rg 1 -sb l:-14,1 -sl u:10,51 -st u:100000,10000000 -si u:1,6
-so l:0,0.1 -gb u:-25,-15 -gt f:gb -ld l:gb,0.4 -lb f:ld -lt l:gt,0.4 -lk 1 -sp l:12,0.5 -sg
f:1 -hs l:1.5,1 -hl l:1.2,1 -hg l:1.4,1 -su e:10000000 -o SimPhy_complex -om 1 -v 1 -od 1
-cs 22
```

6. The SimPhy model

SimPhy implements a hierarchical phylogenetic model in which gene trees evolve inside locus trees, which in turn evolve along a species tree, similarly to the DLCoal model (Rasmussen and Kellis, 2012). In general, we refer to "species" as any diverging, interbreeding group of individuals regardless of its taxonomic rank.

The **species tree** depicts the evolutionary history of the sampled organisms. Its internal nodes represent speciations, while branches reflect the population history between them. Thus, branches inform about the effective population size (Ne, usually represented as branch width), number of generations and time.

The **locus trees** represent the evolutionary history of the loci sampled from a given gene family, and is embedded within the species tree. Internal nodes represent either genetic divergence due to speciation or locus-level events like duplications, losses, horizontal gene transfers or gene conversions, while branches reflect the history of loci populations, and consequently contain information for Ne, number of generations and time, in the same way as the species tree.

The **gene trees** represent the evolutionary history of the sampled gene copies that evolve inside a locus tree. Gene tree nodes indicate coalescent events, while branches represent the expected number of substitutions per site.

This model encompasses seven evolutionary processes defined as follows:

- **Speciation**: separation of one ancestral population into two new populations that do not interbreed.

- **Extinction**: disappearance of a species.
- **Gene duplication**: copy of one gene into a new locus in one individual of the population, which gets fixed in the sample (i.e., we assume no duplication polymorphism).
- **Gene loss**: deletion of one gene in one individual of the population, which gets fixed in the sample (i.e., we assume no loss polymorphism).
- **Horizontal gene transfer**: copy of one locus from one species to another contemporary species via replacement. The transfer initially affects one individual in the receptor species and then gets fixed in the sample (i.e., there is no transfer polymorphism).
- **Gene conversion**: replacement of one homolog by another within a single species. This conversion initially affects one individual and then gets fixed in the sample (i.e., there is no gene conversion polymorphism).
- **Lineage sorting**: consideration of the coalescent process of the sampled gene copies, allowing their history to be incompatible with the species tree history.

All these processes but speciation are considered independently for each gene family.

Our model builds upon the multispecies coalescent model (Rannala and Yang, 2003) inheriting the same assumptions: each branch of the species tree is composed by a perfect Wright-Fisher population (fixed population size, non-overlapping generations, random mating, neutrality), no recombination within loci and free recombination between loci. Moreover, locus tree events (duplication, transfers and conversions) generate coalescent bounds –since they just originally affect one individual– and therefore generate subtrees modeled by the bounded multispecies coalescent (Rasmussen and Kellis, 2012). We assume that these events (together with losses) either get fixed or extinct in the sampled lineages. The locus tree is described by a birth-death process modeling duplication and losses –parameterized by a birth and a death rate per generation– coupled with two pure birth processes that respectively describe the transference of genetic material and gene conversion events –parameterized by their own rate and the state of the birth-death process. Thus, populations with more loci per species branch are more likely to transfer genetic material –thus keeping constant the transfer probability per locus. *SimPhy* incorporates two HGT model variants, one in which the transfer receptor is randomly chosen form the candidates and another that takes into account the evolutionary distance between donors and candidate receptors (reception probability inversely proportional to the distance).

Model levels

Tree	**Nodes**	**Edge Lengths**	**Width**	**Simulation strategy**
Species	Speciations	Elapsed time (generations or time)	Ne	Yule,Birth-Death or user defined
Locus	Speciation,GDL,HGT,GC	Elapsed time (generations or time)	Ne	Birth-Death + 2 Pure-Birth processes

Gene	Coalescences	Exp. nº of subs per site	N/A	Multispecies + Multilocus coalescent

Relaxed clocks

As described above, species, locus and gene trees are ultrametric. In order to relax the molecular clock and perform more realistic simulations, *SimPhy* incorporates different sources of rate heterogeneity: species-specific, locus-specific and gene-by-lineage-specific.

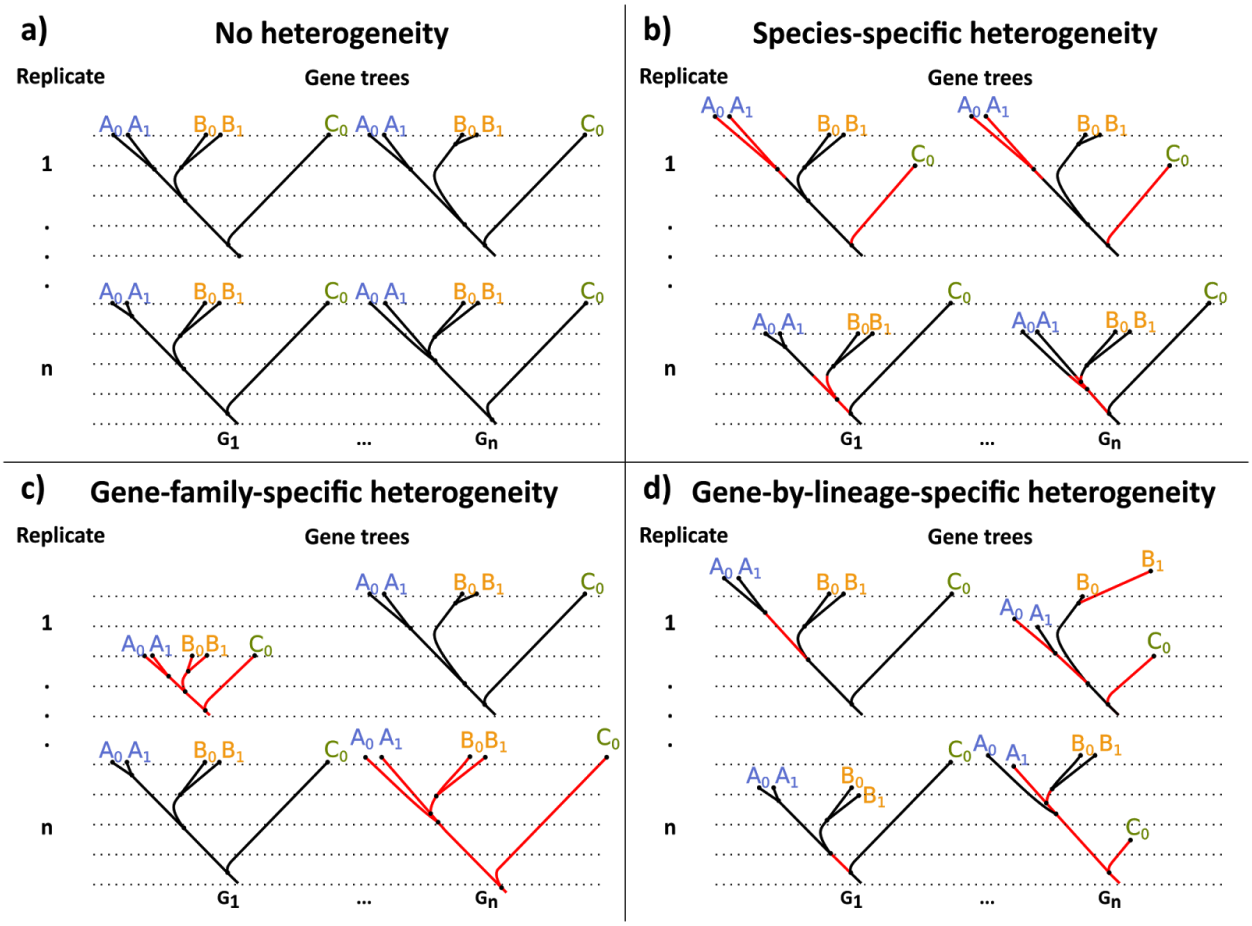

Fig. 2. Sources of substitution rate heterogeneity among lineages available in SimPhy. Each subfigure represents the same simulation scenario but with different types of rate variation. Each simulation scenario consists of two independent replicates of two gene families (columns) in three species (A,B,C) with five individuals (A0,A1,B0,B1,C0). Subfigure a) shows trees simulated without substitution rate heterogeneity –hence ultrametric– with topologies and branch lengths generated by the coalescent process. Subfigure b) depicts a case of species-specific heterogeneity, where changes in rate affect whole ancestral/contemporary species genome-wide (e.g., for every gene family in replicate 1 species A branches evolve faster and species C branches slower). Subfigure c) is an example of gene-family-specific heterogeneity, where rate changes affect the total history of specific gene families (note that in this case ultrametricity still holds) (e.g., in replicate 1, gene family 1 evolves much slower than gene family 2). Subfigure d) shows gene-by-lineage-specific heterogeneity, where

changes in rate affect particular gene tree branches independently (e.g., in replicate 1, gene family 1, the ancestral A lineage evolves faster). In addition, the level of heterogeneity can also be modulated among replicates (e.g., in subfigure c) replicate 1 shows much more heterogeneity than replicate n).

After tree generation the substitution rate of the different branches can be modified. This modification will be reflected only at the gene tree level, since locus and species trees are measured in generations. The different levels of heterogeneity rely on sampling multipliers from Gamma distributions with mean one for every branch of the species tree, for each gene family and for every gene tree branch. The Gamma distribution parameters can be controlled using hyper-parameters and hyper-hyper-parameters that can be linked. The most comprehensive configuration for the case of species-specific and gene-family-specific heterogeneity rely on genome-wide parameters that act as hyper-parameters. Thus, for each species tree two parameters are sampled from their respective genome-wide distribution and then used to describe the Gamma distributions to sample the substitution rate multipliers for each species tree branch and locus tree, respectively. For the case of the gene-branch-specific heterogeneity the scheme is even more complex, since the genome-wide parameter is used as a hyper-hyper-parameter. Thus, the genome-wide distribution is sampled for each species tree, which in turn is used to describe a distribution to sample one parameter per locus tree. Finally, these locus-specific parameters are used as parameters for the Gamma distribution to sample the multipliers for each gene tree branch.

Apart from the substitution rate heterogeneity, when the species trees and locus trees are user-defined, branch-specific effective population sizes and generation times can be specified.

7. Sampling strategy

Species trees are sampled either using a pure birth (Yule) model or a birth-death model, parameterized by the birth (speciation) and death (extinction) rates (events per generation) and either the number of leaves, the tree height or both. The simple sampling approach algorithm (SSA) is used when the number of leaves is not specified, while the birth-death rate approach (BDSA) algorithm is used otherwise (for more details see Hartmann et al. 2010).

Locus trees are simulated using one SSA to sample a birth-death process –respectively describing duplications and losses– coupled with two pure birth processes –describing horizontal gene transfers and gene conversions– across each branch of the species trees, traversed in pre-order. This is followed by a second pre-order traversal that samples receptors from contemporary candidates for each horizontal gene transfer/gene conversion, performing the corresponding SPR branch rearrangement to generate the definitive locus tree.

Gene trees are sampled traversing the locus tree in post-order. Locus tree branches that not pertain to bounded subtrees are modeled by the multispecies coalescent, and therefore sampled by common coalescent simulation strategies –i.e. a rejection sampling strategy where waiting times come from an Exponential distribution. However, branches pertaining

to bounded subtrees imply a much more complex strategy, based on first sampling the number of lineages going into and out of them and then obtaining the coalescent times conditioned on these numbers. This procedure requires the input and output probabilities of lineage counts for every branch of the these subtrees, which are calculated by *SimPhy* using a dynamic programing algorithm modified from the one proposed by Rasmussen and Kellis (2012). A pre-order recursion is then performed to sample the number of lineage counts going across every branch. This is carried out using the inverse transform sampling method with the CDF of the number of input lineages conditioned on the pre-calculated input probabilities, output lineage counts, effective population size and branch length for every branch of the considered subtree (see Mallo et al. 2015 for more details). Hence, it starts at the root of each bounded subtree where the number of output lineages is fixed to one, and then samples the counts along the tree, stopping at the leaves. With the lineage counts already set, the coalescent times are sampled using the inverse transform of the CDF of the coalescent times conditioned on lineage counts.

7.1 Example pipeline

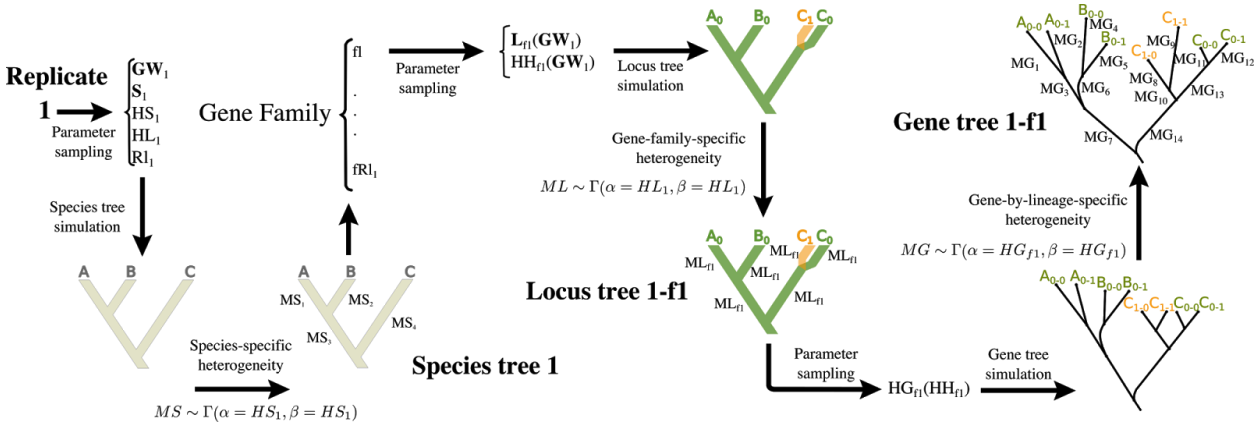

Fig. 3. SimPhy type case flowchart. For simplicity it is only represented one simulation replicate (1) and one gene family (f1). Parameters sampled for replicate 1 before the species tree simulation: vector of genome wide parameters ($\mathbf{GW}_1$), vector of species tree parameters ($\mathbf{S}_1$), species-specific heterogeneity parameter (HS_1), gene-family-specific heterogeneity parameter (HL_1), number of gene families for replicate 1 (Rl_1). After the species tree simulation the species-specific heterogeneity is applied, sampling one substitution rate multiplier per species tree branch (MS). Parameters sampled the gene family f1, prior to the locus tree simulation: vector of locus tree parameters ($\mathbf{L}_{f1}$) and the gene-by-lineage heterogeneity gene family parameter (HH_{f1}), both dependent upon $\mathbf{GW}_1$. After the locus tree simulation the gene-family-specific heterogeneity is applied, sampling one multiplier (ML_{f1}) and applying it for every branch of the tree. The gene-by-lineage parameter (HG_{f1}) is sampled before the gene tree simulation. Finally, the gene-by-lineage heterogeneity is applied, sampling one multiplier (MG) per gene tree branch. **GW**=Duplication parameter, loss parameter, transfer parameter, gene conversion parameter, gene-by-lineage heterogeneity parameter. **S**=Speciation rate, extinction rate, species tree height, number of taxa, outgroup distance, number of individuals per species, tree-wide effective population size, tree-wide substitution rate, tree-wide generation time. **L**=Duplication rate, loss rate, horizontal gene transfer rate, gene conversion rate.

8. References

Fletcher W and Yang Z (2009) INDELible: a flexible simulator of biological sequence evolution. *Mol. Biol. and Evol.* **26**(8): 1879-88

Hartmann K, Wong D and Stadler T (2010) Sampling trees from evolutionary models. *Syst Biol* **59**: 465-76

Rannala B and Yang Z (2003) Bayes estimation of species divergence times and ancestral population sizes using DNA sequences from multiple loci. *Genetics* **164**: 1645-56

Rasmussen MD and Kellis M (2012) Unified modeling of gene duplication, loss, and coalescence using a locus tree. *Genome Research* **22**: 755-65

Tange O (2011) GNU Parallel - The Command-Line Power Tool. *;login: The USENIX Magazine* **36**(1): 42-47