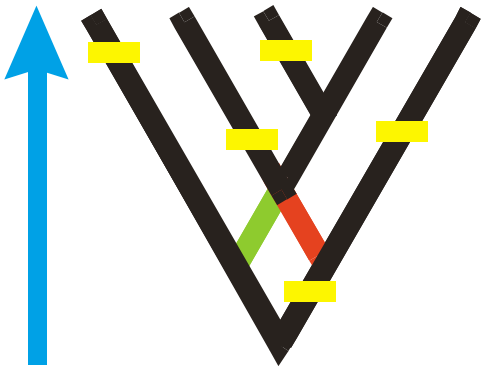
Documentation for **RECODON**

Coalescent simulations of codon models with recombination, migration and demography

Current version is 1.6.0

© 2006-2009 Miguel Arenas (miguelab@uvigo.es) and David Posada (dposada@uvigo.es).

November 19, 2009



Reference [1]: ARENAS, M., and D. POSADA, 2007 Recodon: coalescent simulation of coding DNA sequences with recombination, migration and demography. BMC Bioinformatics **8:** 458.

Contents

Disclaimer

Credits

This program was developed at the Department of Biochemistry, Genetics and Immunology of the University of Vigo (Spain).

History

Version 0.1-0.2

- *Program Recoal by D. Posada: coalescent with recombination keeping trees for each sites. Simulates GTR+I+G and nested models.*

Version 0.3 (September 2006)

- Simulates genealogies under the coalescent with recombination keeping trees for segments.
- Included fluctuating effective population size and exponential growth functions from SNPsim [2].
- Compute both *TGMRCA* and *TMRCA*. All recombinant fragments have the same *TGMRCA* but also can have different *TMRCA*.

Version 0.4 (October 2006)

- Allow for several demographics periods.
- *MRCA* and *GMRCA* times average in each replicate and between all replicates.

Version 0.5 (November 2006)

- User can introduce the *MRCA* sequence from an input file.
- Other updates.

Version 0.6 (January 2007)

- Implemented *Goldman & Yang 1994* codon model crossed with GTR and with frequencies given by the nucleotides frequencies at each codon position: *GY94xREV_3x4*.
- Note that breakpoints occur between codons.

Version 0.7 (March 2007)

- Implemented a finite island model.
- Variance of *GMRCA* and *MRCA* times for all replicates.

Version 0.8 (July 2007)

- Sites update.
- Print separated sequences about an output file per replicate.
- No stop codons in codon models. Update codon models.

Version 0.9 (August 2007)

- Omega variable among sites by categories.

Version 1.0 (September 2007)

- Variable omegas can be represented by user-specified categories, or by a discrete or continuous gamma distribution.
- Added non reversible models
- Minor fixes.

Version 1.1 (October 2007)

- Update code. Some bugs were corrected.
- Starting MPI version. A lot of Thanks to CESGA (Centro de Supercomputacion de Galicia) and specially to J. Carlos Mouriño Gallego.

Version 1.2 (October 2007)

- Print ancestal sequences only when there is no recombination, as then we do not the complete sequence of the internal nodes, except for the root.
- Improved run summary.
- Print replicate number with noisy 0.
- The argument to introduce the nucleotide frequencies was improved. In this version, the first number after the "-f" argument has to be the number of nucleotide frequencies (4 for nucleotide and codon models or 12 only for codon models), and the next numbers are the value of the nucleotide frequencies.
- Improve of screen files for MPI ejecution (the functions to build the Q matrix and to calculate gammas to a variable omega, do not write in these screen files).
- The number of processors cannot be bigger than the number of replicates for MPI ejecutions.

Version 1.3 (November 2007)

- Minor fixes.
- Choosing susbtitution model in "main" function, after introduce the input parameters.
- Improve codon models. For eigen functions, the number of digits in codon models must to be different to the number of digits in nucleotide models.
- Improve of the "fixed rec events code". In MPI version each processor runs only one replicate and the number of replicates must to be constant to distribute them to the processors. Before this version, the number of replicates does not constant when rec events was fixed. This was two problems for MPI running, the number of replicates was no costant to distribute them to the processors and the seed was the same for this replicate, so the replicate repeated itself and it never ends. In this version, when rec events is fixed, "subreplicates" run in one replicate, and this replicate ends when a subreplicate is accepted, this solves all the problems. This happens now for MPI and no MPI executions.
- Improve print for MPI and lineal versions (using fpmpi).

Version 1.4 (January 2008)

- Evolution of demes by events of convergence of demes. Two demes can converge to a single deme in a forced event from the user. The migration events are conditionated by the events of demes convergence.
- Improved minor fixes.
- Option to print the settings in an output file (active in code).
- Bug solved about initial memory when there were very high population size and very high recombination rate.

Version 1.5 (June 2008)

- Minor fixes.

1.5.1 (October 2008)

- Print sequences for phylip output files with migration in style: "s00001_p1.. and outgrp_p0".
- Print output sequences in fasta format.

1.5.2 (October 2008)

- All the codon models with omega variable have the omega variable per codon, not per branch. Improved codon models with omega variable per codon.
- Option for haploid/diploid from user.
- Print alignments in nexus format.

Version 1.6 (January 2009)

- M1, M7 and M8 codon models.
- Print file with dN/dS simulated for each site.

1. Purpose

Recodon [1] generates samples of nucleotide and codon sequences from populations with recombination, migration and demography, according to the coalescent [3-5].

2. Executables and compilation

Executable files are provided for Linux Debian, MacOS X (Intel and G5 processors) and Windows XP, and one Makefile is provided for compilation in any OS with a C compiler. This makefile can be optimized for different users, for example using the optimization option –fast instead of –O3, for G5 Mac processors.

A second makefile "Makefile_MPI" is provided to compile a MPI version. This makefile might need some modifications for particular OS. To compile the program type: *make -f Makefile_MPI*

MPI libraries have to be installed in the host for running *Recodon* in parallel. The minimum number of processors is two. An example of execution for 3 processors is the next: *mpirun -np 3 recodon1.6.0*

3. Recodon Usage

The input of the program consists of a series of arguments and parameter values (Table 1) that can be entered in the command line or, more conveniently, specified in a text file

called "parameters" that should be located in the same directory as the executable. These arguments include the parameter values used in the simulations and several printing options that control the amount of information that is sent to the console or output files.

3.1. Command line

In Mac systems, the command line is provided by the Terminal, while in Windows XP, this is provided by the windows console or command prompt. If the user specifies any argument in the command line, *Recodon* will use the values specified for those parameters, and default values for the parameters not included in the command line. It is very important that the user checks the screen info to make sure that the simulations are being parameterized as intended. Some example command lines are:

./Recodon1.6.0 -n10 -s6 –l300 –e200

./Recodon1.6.0 -n10 -s6 –l101 -e1000 -p3 1000 1250 1000 1300 1550 2000 1560 1000 3000 -r1.0e-06 -u0.0015 -q2 3 3 0.00125 -%1 1 2 4000 –m2 3 1.2 0.2 0.4 0.3 1.3 0.5 -v1.0 1.0 1.0 1.0 1.0 1.0 –f12 0.23 0.21 0.33 0.23 0.23 0.21 0.33 0.23 0.23 0.21 0.33 0.23 -i0.5 –a0.5 -xseqMRCA -bsequences -jtrees -ktimes -dbreakpoints -c –z -o0.1 -y4 -#386658297

3.2. Parameter file

If no argument is specified, the program will look for the text file "*parameters"* in the same directory as the executable. If no arguments are specified in the command line, and there is not "*parameters*" file, the program will stop and throw an error. I*n this file* anything within brackets will be ignored. An example parameter file is included in the distribution.

3.3. MRCA input file

By default, the most recent common ancestor (MRCA) sequence is simulated according to the nucleotide frequencies. However, the user can specify its own MRCA sequence from a text file, which has to be located in the same directory as the program.

3.4. Output Files

All output files produced by the program are written to the folder *Results*. There are four optional output files

- *Sequences file*: contains the aligned sequences in Phylip sequential, Nexus and Fasta formats.
- *Breakpoints file*: contains the breakpoints ordered by event time.
- *Trees file*: contains the trees in Newick format for each recombinant fragment.
- *Times file*: contains time and branch length info for each tree.
- *Simulate omegas per site*: contains the simulated omega value per codon. This option can be only used for codon models with variable omega per site.
- *Settings file*: contains the final settings. This option must be active in the source code.

3.5. Arguments

Possible arguments for *Recodon* are (# = number):

-n# : Number of replicates
The number of samples to be generated. Each sample is an independent realization of the evolutionary process. Example: *-n10*

-s# : Sample size
The number of sequences to be generated for each sample. Example: -s6

-l# : Number of sites
The total length, in nucleotides or codons, of the sequences. Example: *-l300*

-e# : Effective population size
The effective size (*N*) of the population from which the sample was theoretically drawn. If there are several demes, this argument is the effective population size for each deme. Example: *-e100*

-_# : Haploid / Diploid
Data set can be simulated as haploid or diploid. Example: *-_2*

-g# : Exponential growth rate
Rate of exponential growth per individual per generation. This option is incompatible with the demographic periods (*-p*), see below. Note: these parameters are looking back in time, so it is not a good idea to specify a negative growth rate for the last period, as the coalescent time could become infinite in the past. Example: *-g1e-5* (= *-g0.00001*)

-p# (# # #) : Demographic periods
The number of periods (from present to past) and *N* during those periods. The first number specifies the number of periods. For each period there should be three consecutive numbers indicating the size *N* at the beginning and at the end of the period, and the duration of the period in generations. Example: *-p3 1000 1250 1000 1300 1550 2000 1560 1000 3000*

The exponential growth rate during the period (positive or negative) will be deduced from the specified *N* at the beginning and at the end of the period. The growth rate derived for the last period will continue into the indefinite past. This implementation is borrowed from [5]. This option is incompatible with the exponential growth rate option (*-g*). Again, these parameters are looking back in time, so it is not a good idea to specify a negative growth rate for the last period, as the coalescent time could become infinite in the past.

-r# : Recombination rate
The recombination rate per site (nucleotides or codons) per generation. All sites will share this same rate. Example: *-r2e-6* (= *-r0.000002*)

-w# : Fixed number of recombinations

This option fixes the number of recombinations per replicate, so every sample will have the same number. What it does is to filter out replicates with a different number of recombination events, so it will take more time. Example: *-w2*

-q# (#) # : Migration parameters
The first number specifies the total number of demes or subpopulations sampled. The next *n* numbers specify the number of individuals (or sequences) per deme (note that the specified sample size (*-s*) must to be equal to the sum of these). The last number is the migration rate per individual per generation. Example: -q2 3 3 0.00125 (two demes with three individuals each, with a migration rate of 0.00125.

-%# (# # #) : Events of convergence of demes
The first number specifies the total number of convergent events. For each convergence event there should be three consecutive numbers. The first number and the second number are the numbers of the demes to converge. The third number is the time to that convergence. With this option the user can build the evolutionary tree of the demes and it is just available when the migration model is running (despite the migration rate can be zero). Examples: -%1 1 2 2000 (for 2 initial demes (-q2), convergence of deme 1 with deme 2 at time 2000 to create a new deme 3). -%3 1 2 400 3 4 1900 5 6 2000 (for 4 initial demes (-q4) convergence of deme 1 with deme 2 at time 400 to create a new deme 5, convergence of deme 3 with deme 4 to create a new deme 6 at time 1900, convergence of deme 5 with deme 6 at time 2000 to create a final new deme 7).

-u# : Mutation rate
Mutation rate per site per generation. Example: *-u0.0015*

-f# # # # # : Nucleotide frequencies
The nucleotide frequencies A C G T are specified in this order. It is possible introduce 4 or 12 frequencies. When 4 frequencies are introduced in codon models the values of the frequencies for all codon positions are the same. Only for codon models, the frequencies of A_p C_p G_p T_p can be specified in turn for each of the three positions (3×4). By default all frequencies are equal. Four frequencies example: *-f4 0.4 0.2 0.1 0.3*. Twelve frequencies example: *-f12 0.3 0.2 0.4 0.1 0.2 0.2 0.3 0.3 0.2 0.2 0.1 0.5*

-t# : transition/transversion rate ratio
Transition/transversion rate ratio. When set to 0.5, the probability of transitions and transversions are the same, like in the models "*JC*" [6] or "*F81*" [7] models. Example: *-t2.1*

-v# # # # # # # : relative symmetric substitution rates
Relative substitution rates A↔C, A↔G, A↔T, C↔G, C↔T and G↔T. Example: *v1.0 5.0 1.0 1.0 5.0 1.0*

-@# # # # # # # # # # # # #: relative asymmetric substitution rates
Relative substitution rates AC, CA, AG, GA, AT, TA, CG, GC, CT, TC, GT and TG. Note than some calculations under this option can be problematic (complex eigenvalues/vectors) if rates are too asymmetric, specially with codon models. Example: *@1.1 1.2 1.3 1.4 1.5 1.6 1.7 1.8 1.9 2.0 1.0 1.0*

-m# #: nonsynonymous/synonymous rate ratio (ω)

This options control the implementation of the nonsynonymous/synonymous rate ratio ($\omega = d_N/d_S$) for codon models. Four different models can be specified:

- Model 1 (-m1): ω is constant across sites. The first number specifies the model, and the second number is the value of ω. Example –m1 1.63

- Model 2 (-m2): ω varies across sites according to some user-defined categories. The first number specifies the model and the second number is the number of categories. The next numbers are, for every category, the omega value and its corresponding probability. Example –m2 3 1.2 0.2 0.4 0.3 1.3 0.5

- Model 3 (-m3): ω varies across sites according to a discrete gamma distribution [8]. The first number indicates the model, and the second number is the number of equally probable categories (no more than 10) for the gamma distribution. The third number is the alpha shape of the gamma distribution and the last number is the average ω. Example –m3 3 0.78 1.4

- Model 4 (-m4): ω varies across sites according to a continuous gamma distribution [8]. The first number indicates the model. The second number is the alpha shape of the gamma distribution and the third number is the average ω. Note that this model can be very slow. Example –m4 0.91 1.42

- Model 5 (-m5): M1 codon model. Two categories ($\omega 0$ *P0,* $\omega 1$ *P1*), two values of ω with a proportion each category. The first number specifies the model, the second number is the proportion of the $\omega 0$ *(P0*), and the third number means the value of $\omega 0$. Remember that for M1, $\omega 1$ = 1.0 and *P1=1-P0.* Example –m5 0.8 0.9

- Model 6 (-m6): M7 codon model. ω varies across sites according to a beta distribution. The first number specifies the model, the second number is "*p*" and the last number is "*q*" of the beta distribution. Example –m6 1.4 0.5

- Model 7 (-m7): M8 codon model. ω varies across sites according to two categories, one is a beta distribution and the other is a ω fixed for the user. The first number specifies the model, the three next numbers are for the category of the beta distribution, the *P0* (proportion the sites to evolve by the beta distribution), "*p*" and "*q*" for the beta distribution. Finally, the last number is the $\omega 1$ fixed for the user for the second category. Remember that *P1=1-P0*. Example –m7 0.05 5.01 3.05 2.00

-a# : alpha shape of the gamma distribution

A gamma distribution (+*G*) is used to simulate substitution rate variation among sites [8]. Alpha is the shape of this distribution. Smaller alphas imply stronger rate variation. Example: *-a0.5*

-i# : proportion of invariable sites

A proportion of sites can be set to be invariable (+ *I*). Example: *-i0.5*

-o# : outgroup branch length

If this option is specified the program simulates an outgroup sequence that evolves independently from the sample, along a branch of the specified length. By default the outgroup is not simulated. Example: *-o0.1*

-xNAME : MRCA input file

The user can specify its own MRCA sequence in a text file. This file must contain only a DNA sequence, which length has to be equal to the number of sites specified. By default, the *MRCA* is simulated from the nucleotide frequencies (see above, -f argument). Note that the MRCA sequence actually represents the concatenated

sequence of the different MRCAs for the resulting recombinant fragments. Example: *-xseqMRCA*

-bNAME : print sequences
When this argument is invoked, aligned sequences are printed to the specified text file in the *Results* folder, in Phylip sequential format. Example: *-bdatafile*

-c : print ancestral states
This option will print the ancestral sequence of all internal nodes only in the absence of recombination. If there is recombination, many internal nodes will contain non-ancestral material, and therefore in this case only the sequence of the MRCA will be printed Example: *-c*

-z : multiple alignment files
When this argument is invoked, each alignment for each replicate is printed to a different file. Example: *-z*

-# : format of print sequences*
This option specifies the format of the output alignments, it is possible by Phylip sequential, Fasta and Nexus. According to the number: 1 generates phylip sequential format, 2 generates fasta format and 3 generates nexus format. Example: *-*2*

-jNAME : print genealogies
When this option is specified, genealogies for each recombinant fragment are printed, in Newick format, to the specified text file in the *Results* folder. Example: *-jtrees*

-kNAME : print times
When this argument is specified the coalescent times for each genealogies will be printed to the specified text file in the *Results* folder. This option will slow down the simulations. Example: *-ktimes*

-dNAME : print breakpoints
When this argument is specified breakpoints positions are printed to the specified text file in the *Results* folder. Example: *-dbreakpoints*

-+ : print Simulate omegas per site
When this argument is specified the omegas simulated per codon are printed to the specified output file in the *Results* folder. Example: -+

-y# : noisy level
This option controls the level of information that will be printed to the screen.

0 : does not print run information, just the simulation progress.

1 : run settings and run information summarizing the simulations.

2 : calculation status, initial demes and event information, run settings for each sample

3 : print ancestral status for each sequence at each event + *MRCA* status

4 : potential recombining locations (*g* and *G* vectors) and information about recombinant fragments evolution.

Note that higher levels of noisy will slow down the simulations. The default level is 1 Example: *-y1*

-## : Seed
Seed for the random number generator. If no seed is specified, the computer clock will be used. Example: *-#386658297*

3.6. Default settings

By default *Recodon* will simulate 10 samples of 6 individuals with 201 sites, a constant effective size of 1000, constant size, no recombination, no migration and a mutation rate of 1e-7 under the *J-C* nucleotide model with each nucleotide frequency value equal to 0.25. Noisy level is 1. The sequences will be printed to the *sequences* output file in the *Results* folder. To run the program with the equivalent arguments we would type:

./Recodon1.6.0 –n10 –s6 –l201 –e1000 –p0 –r0.00 –q1 6 0.00 –u1.0e-07 –f4 0.25 0.25 0.25 0.25 –t0.5 –a0.00 –i0.00 –bsequences –y1

Table 1. Key arguments for Recodon. The user can specify several parameters to define different simulation scenarios. These arguments can be entered in the command line or read from text file.

Parameter	Example value	Application
Number of replicates	1000	All
Sample size	12	All
Number of sites (bp or codons)	3000	All
Effective population size	1000	All
Exponential growth rate	$2.1x10^{-5}$	Demography
Demographic periods[1]	1000 5000 200	Demography
Recombination rate	5×10^{-6}	Recombination
Migration rate	1.2×10^{-4}	Migration
Number of demes	4	Migration
Events of convergence of demes[2]	1 2 5000	Demes/Migration
Mutation rate	5.1×10^{-4}	All
Nucleotide frequencies[3]	0.4 0.3 0.1 0.2	Nuc/codon models
Transition / transversion ratio	2.1	Nuc/codon models
Relative substitution rates	1.0 2.3 2.1 3.0 4.2 1.0	Nuc/codon models
Nonsynonymous / synonymous rate ratio[4]	1.8	Codon models
Rate variation among sites[5]	0.5	Nuc/codon models
Proportion of invariable sites	0.2	Nuc/codon models
Print simulate omegas per site	+	Codon models with variable omega per site.
Format sequences output	2	All

[1]from 1000 to 5000 effective size during 200 generations.

[2]deme 1 is converging with deme 2 at time 5000 to make a new ancestral deme 3.
[3]can be specified for each codon position in codon models (3×4).
[4]*dN/dS*. There are four options to implement to implement this parameter.
[5]shape of the gamma distribution.

4. Recodon model

The model implemented in *Recodon* is an extension of the coalescent with recombination based on the neutral Wright-Fisher model [4, 5]. Given the specified recombination and migration rates and other parameters like the effective population size (N) and growth rate, random genealogies are produced. The evolution of the demes can be fixed. Complex demographic histories can be implemented by defining demographic periods in which population sizes augment, reduce, or remain constant. Mutations can be placed upon the genealogies under nucleotide or codon models. The result is a random sample of aligned nucleotide or codon sequences.

The coalescent proceeds backwards starting from the sample of s gametes. Time is scaled in units of $2N$ generations, where N is the effective population size. For a given site, without recombination or migration, and under constant population size, the time to the most recent common ancestor (TMRCA) is:

$$E(TMRCA) = 2\left(1-\frac{1}{s}\right)$$

$$Var(TMRCA) = \sum_{i=2}^{s}\frac{4}{i^2(i-1)^2}$$

The times to a coalescence (CA), recombination (RE) or migration (MI) event are exponentially distributed, with intensity equal to their respective rates (see below). The next event will be the one that would occur before according to these expectations.

$$Time\ to\ CA \propto Exp\left[rateCA\right]\cdot 2N$$

$$Time\ to\ RE \propto Exp\left[rateRE\right]\cdot 2N$$

$$Time\ to\ MI \propto Exp\left[rateMI\right]\cdot 2N$$

4.1. Coalescence

The rate of coalescence depends only on the number of lineages (k):

$$rateCA = \frac{k(k-1)}{2}$$

4.2. Recombination

The rate of recombination depends on the population size (N) and on the total recombination rate at all valid recombination sites (G). A *valid* recombination site has to have at both sides ancestral material that has not found its MRCA yet. In codon models, recombination can only occur between codons.

$$G = \sum_{j=1}^{k} \sum_{i=1}^{L} r_{Gi}$$

$$\rho = rateRE = 2NG$$

Given that a recombination event occurs, a gamete is chosen according to the total rate at potential recombining sites in that gamete. Breakpoint sites are chosen according to the recombination probabilities per site (r_{Gi}). The expected number of recombination events in a panmictic population with constant size is:

$$E(number\ of\ recombination\ events) = \rho \sum_{i=1}^{s-1} \frac{1}{i}$$

Importantly, in the presence of recombination, different regions of the alignment might evolve under different genealogies (Figure 1), which together conform to the ancestral recombination graph. The number of genealogies will be the number of breakpoints + 1.

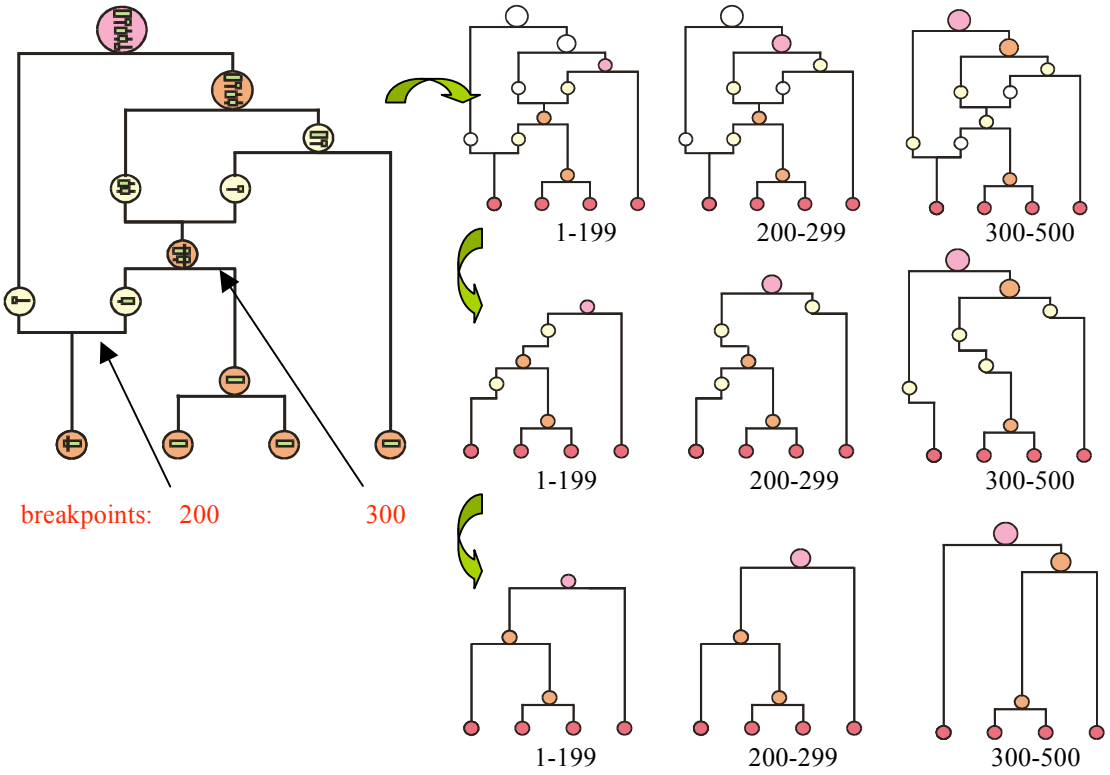


Figure 1. Representation of the ancestral recombination graph and the binary tree embedded.

4.3. Migration

The simplest and most widely used model of population structure is the " finite island model" [9-13]. This describes an array of *q* demes, each of constant size (Figure 32). The rate of migration depends on the population size (*N*), the migration rate per deme (*m*) and the number of demes (*q*).

$$rateMI = 2Nmq$$

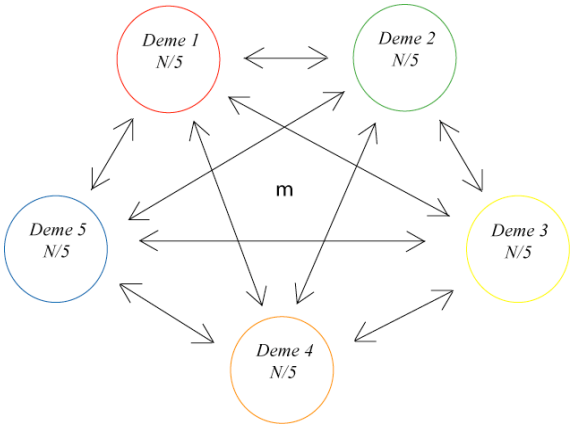


Figure 2. Island model. All demes have the same population size and the same migrant rate per deme.

In the coalescent with migration, lineages can only coalesce with other lineages in the same deme. When a migration event occurs, a given lineage changes from one deme to another (Figure 3).

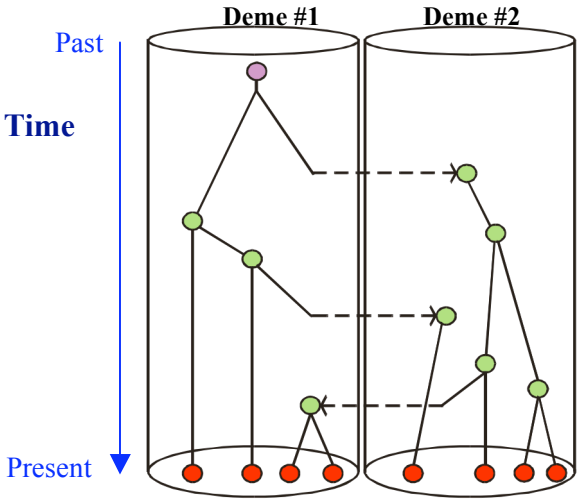


Figure 3. Coalescent with migration for two demes.

4.4. Convergence of demes

A new tool have been implemented in *Recodon* to fix the evolution of the demes. Two active demes can be converged at a given time to build a new big deme with the nodes of the previous demes. The convergence of demes always success with a migration model although the migration rate can be zero. An example about events of the convergence of demes is showed in the figure 4.

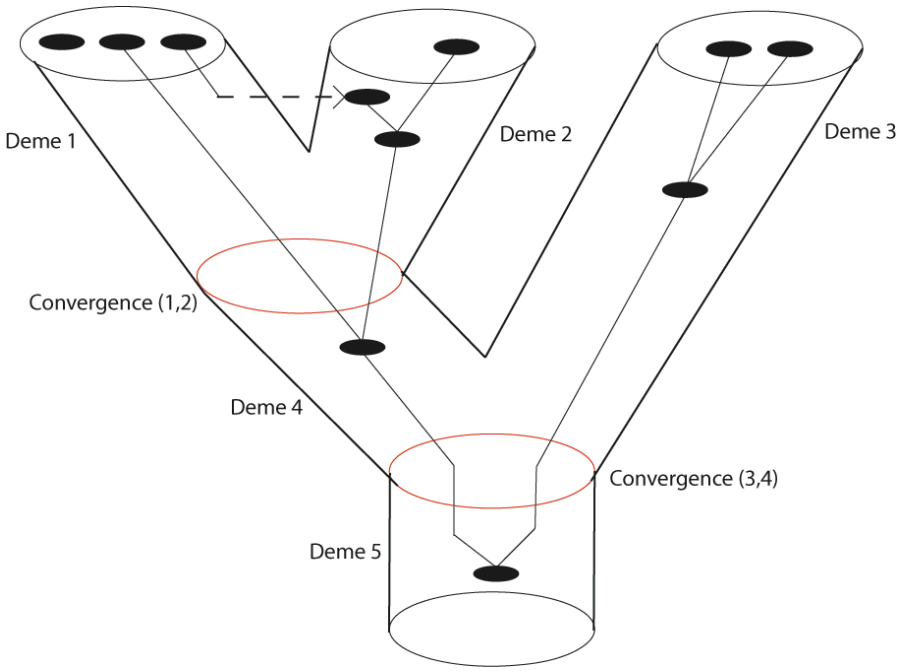


Figure 4. Demes evolution. Two demes can evolve convergences to generate an ancestral big deme, these convergences are introduced by the user.

4.5. Demography

Recodon allows for the specification of exponential population growth rate. The only modification concerns to the expected time to a coalescence, where t is the current time:

$$time\ to\ CA \sim \frac{\log\left[e^{\beta t} + \beta Exp\left[\frac{k(k-1)}{2}\right]2N\right]}{\beta} - t$$

If the growth rate is negative, the coalescence time may be infinite (i.e., coalescence does not happen), and *Recodon* will stop and issue an error message. Alternatively, the user can define any number of demographic periods (Figure 5). β_i is the growth rate inferred for a demographic period i that goes from size N_{Bi} in the past to size N_{Ei} in I_i generations:

$$\beta_i = \frac{-\log\left(\frac{N_{Bi}}{N_{Ei}}\right)}{l_i}$$

The time to coalescence will be:

$$\text{Time to CA} \sim \frac{\log\left[Exp\left(\frac{k(k-1)}{2}\right)\beta_i 2N_{Ei}e^{-\beta(t-t_{i-1})}+1\right]}{\beta_i}$$

where t is the current time and t_i is the cumulative time from the present:

$$t_i = \sum_{j=0}^{j=i} l_j$$

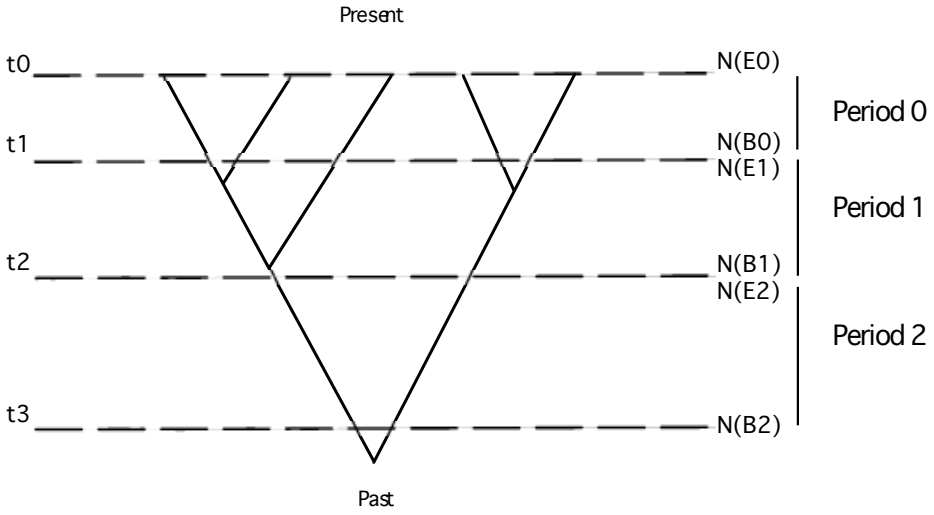


Figure 5. Demographic periods. The growth rate after the last period will the same as the one implied by the last period.

4.6. Substitution models

Once the sample genealogy has been constructed, nucleotide or codon data can be simulated along its branches. Recodon implements several nucleotide and codon models through the specification of different parameters, including base frequencies, relative substitution rates, transition/transversion ratio, nonsynonymous/synonymous rate ratio, a proportion of invariable sites and rate variation among sites. Conveniently, the sequence of the MRCA can be specified at random according to the nucleotide frequencies, or the user can specify its own sequence.

4.6.1. Nucleotide models

Recodon implements the general time reversible model for nucleotide substitution (GTR) [14], a non reversible version (GTnR), and models nested therein, like JC [6], K80 [15], F81 [7] or HKY [16].

4.6.2. Codon models

Recodon implements the GY94 codon model [17], crossed with GTnR, GTR or HKY, and with codon frequencies calculated from the nucleotide frequencies at each codon position (3×4): GY94×GTR_3×4 and GY94×HKY_3×4. A key parameter of the codon models is the nonsynonymous/synonymous rate ratio (ω or *dN/dS)*, that reflects the type of selection acting at the molecular level:

$\omega > 1$: positive selection
$\omega = 1$: neutral evolution
$\omega < 1$: negative selection

NetRecodon implements this parameter in seven different ways.

- Constant, M0: ω is constant across sites.
- Discrete user- categories: ω varies across sites according to some user-defined categories.
- Discrete gamma: ω varies across sites according to a discrete gamma distribution.
- Continuous gamma: ω varies across sites according to a continuous gamma distribution.
- M1 [18]: Two categories of ω value. In the first one ω is < 1, and in the second one ω is fixed = 1.
- M7 [18]: ω varies across sites according to a continuous beta distribution.
- M8 [18]: Two categories of ω value. At the first one ω varies across sites according to a continuous beta distribution, and at the second one the ω is introduced from the user and use to be = 1 or > 1.

The program assumes the universal genetic code without stop codons (i.e., a 61 × 61 matrix). Note that codon models imply much more computation time than nucleotide models.

5. Program benchmarking

Recodon has been validated in several ways. The output of the program was contrasted with the theoretical expectations for the mean and variances for different values, like the number of recombination and migration events or TMRCA, see [19]. Results obtained with Recodon and other programs [5] were compared under different, complex scenarios. Finally, substitution and codon model parameters were estimated from the simulated data using HYPHY [20] and PAUP* [21].

6. Acknowledgments

This work was supported in part by grant BFU2004-02700 (MCyT) to DP and by the FPI fellowship BES-2005-9151 (MEC) to MA from the Spanish government. Several

functions were taken from code provided by R. Nielsen and Z. Yang. We want to thank J. Carlos Mouriño at the Supercomputing Center of Galicia (CESGA) for extensive help with code parallelization.

7. References

1. Arenas M, Posada D: **Recodon: coalescent simulation of coding DNA sequences with recombination, migration and demography**. *BMC Bioinformatics* 2007, **8**:458.
2. Posada D, Wiuf C: **Simulating haplotype blocks in the human genome**. *Bioinformatics* 2003, **19**(2):289-290.
3. Kingman JFC: **The coalescent**. *Stochastic Processes and their Applications* 1982, **13**:235-248.
4. Hudson RR: **Properties of a neutral allele model with intragenic recombination**. *Theor Popul Biol* 1983, **23**:183-201.
5. Hudson RR: **Generating samples under a Wright-Fisher neutral model of genetic variation**. *Bioinformatics* 2002, **18**(2):337-338.
6. Jukes TH, Cantor CR: **Evolution of protein molecules**. In: *Mammalian Protein Metabolism.* Edited by Munro HM. New York, NY: Academic Press; 1969: 21-132.
7. Felsenstein J: **Evolutionary trees from DNA sequences: A maximum likelihood approach**. *J Mol Evol* 1981, **17**:368-376.
8. Yang Z: **Among-site rate variation and its impact on phylogenetic analysis**. *Trends Ecol Evol* 1996, **11**(9):367-372.
9. Wright S: **Evolution in Mendelian populations**. *Genetics* 1931, **16**:97-159.
10. Latter BD: **The island model of population differentiation: a general solution**. *Genetics* 1973, **73**(1):147-157.
11. Maruyama T: **A simple proof that certain quantities are independent of the geographical structure of population**. *Theor Popul Biol* 1974, **5**(2):148-154.
12. Hudson RR: **Island models and the coalescent process**. *Mol Ecol* 1998, **7**:413-418.
13. Matsen FA, Wakeley J: **Convergence to the island-model coalescent process in populations with restricted migration**. *Genetics* 2006, **172**(1):701-708.
14. Tavaré S: **Some probabilistic and statistical problems in the analysis of DNA sequences**. In: *Some mathematical questions in biology - DNA sequence analysis.* Edited by Miura RM, vol. 17. Providence, RI: Amer. Math. Soc.; 1986: 57-86.
15. Kimura M: **A simple method for estimating evolutionary rate of base substitutions through comparative studies of nucleotide sequences**. *J Mol Evol* 1980, **16**:111-120.
16. Hasegawa M, Kishino K, Yano T: **Dating the human-ape splitting by a molecular clock of mitochondrial DNA**. *J Mol Evol* 1985, **22**:160-174.
17. Goldman N, Yang Z: **A codon-based model of nucleotide substitution for protein-coding DNA sequences**. *Mol Biol Evol* 1994, **11**(5):725-736.
18. Yang Z, Nielsen R, Goldman N, Pedersen A-MK: **Codon-substitution models for heterogeneous selection pressure at amino acid sites**. *Genetics* 2000, **155**:431-449.

19. Hudson RR: **Gene genealogies and the coalescent process**. *Oxf Surv Evol Biol* 1990, **7**:1-44.
20. Kosakovsky Pond SL, Frost SD, Muse SV: **HYPHY: Hypothesis testing using phylogenies**. *Bioinformatics* 2005, **21**:676-679.
21. Swofford DL: **PAUP*: Phylogenetic Analysis Using Parsimony (*and Other Methods).** In., 4 edn. Sunderland, Massachusetts: Sinauer Associates; 2000.