User manual for pyvolve v1.0

Stephanie J. Spielman

Contents

1 Introduction

Pyvolve (pronouced "pie-volve") is an open-source python module for simulating genetic data along a phylogeny according to Markov models of sequence evolution. The module is available for download on github (and see here for API documentation). Note that pyvolve has several dependencies, including BioPython, NumPy, and SciPy. These modules must be properly installed and in your python path for pyvolve to work properly. Please file any and all bug reports on the github repository Issues section.

Pyvolve is written such that it can be seemlessly incorporated into your python pipelines without having to interface with external software platforms. However, please note that for extremely large (>1000 taxa) and/or extremelely heterogenous simulations (e.g. where each site evolves according to a unique evolutionary model), pyvolve may be quite slow and thus may take several minutes to run. Faster sequence simulators you may find useful include (but are certainly not limited to!) Indelible [1] and indel-Seq-Gen [8].

Pyvolve supports a variety of evolutionary models, including the following:

- Nucleotide Models
  - Generalized time-reversible model [9] and all nested variants
- Amino-acid exchangeability models
  - JTT [4], WAG [10], and LG [6]

- Codon models
  - Mechanistic (dN/dS) models (MG-style [7] and GY-style [2])
  - Empirical codon model [5]
- Mutation-selection models
  - Halpern-Bruno model [3], implemented for codons and nucleotides

Note that it is also possible to specify custom matrices (detailed in section 7 below). Both site-wise and temporal (branch) heterogeneity are supported. Sequences are simulated accordingy to standard methods [11].

2 Basic Usage

Similar to other simulation platforms, pyvolve evolves sequences in groups of **partitions**. Each partition has an associated size and model (or set of models, if branch heterogeneity is desired). All partitions will evolve according to the same phylogeny; if you wish to have each partition evolve according to a distinct phylogeny, I recommend performing several simulations and then merging the resulting alignments in the post-processing stage.

Pseudocode for a simple simulation is given below.

```
# Import the pyvolve module
import pyvolve

# Read in tree along which pyvolve should simulate
my_tree = pyvolve.read_tree(file = 'file_with_tree_for_simulating.tre')

# Define and construct evolutionary models
my_model = pyvolve.Model(<model_type>, <custom_model_parameters>)
my_model.construct_model()

# Define partitions
my_partition = pyvolve.Partition(models = my_model, size = 100)

# Evolve partitions with the callable Evolver() class
pyvolve.Evolver(tree = my_tree, partitions = my_partition)()
```

2.1 Nucleotide Models

2.2 Amino-acid models

2.3 Codon models

2.4 Mutation-selection models

3 Site-wise heterogeneity

3.1 Nucleotide and amino-acid Models

3.2 Codon models

3.3 Mutation-selection models

4 Temporal heterogeneity

5 Building a vector of stationary frequencies

6 Matrix scaling options

7 Using custom rate matrices

This is my first python example:

```
from pyvolve import *

# Read in a newick tree
t = read_tree(file = "myfile.tre")

# Construct state frequency vector. Optional!
f = EqualFrequencies("amino")
freqs = f.construct_frequencies(type = "codon")

# Build the evolutionary model
m = Model("GY94", {'state_freqs':freqs, 'omega':1.5, kappa:3.4}
m.construct_model()

# Initialize partitions
p = Partition(models = m, size = 100)

# Evolve, and call.
Evolver(partitions = p, tree = t, seqfile = "sequences.phy", seqfmt = "phylip")()
```

References

[1] W Fletcher and Z Yang. INDELible: A Flexible Simulator of Biological Sequence Evolution. *Mol Biol Evol*, 26(8):1879–1888, 2009.

[2] N Goldman and Z Yang. A codon-based model of nucleotide substitution for protein-coding DNA sequences. *Mol Biol Evol*, 11:725–736, 1994.

[3] AL Halpern and WJ Bruno. Evolutionary distances for protein-coding sequences: modeling site-specific residue frequencies. *Mol Biol Evol*, 15:910–917, 1998.

[4] DT Jones, WR Taylor, and JM Thornton. The rapid generation of mutation data matrices from protein sequences. *CABIOS*, 8:275–282, 1992.

[5] C. Kosiol, I. Holmes, and N. Goldman. An empirical codon model for protein sequence evolution. *Mol Biol Evol*, 24:1464 – 1479, 2007.

[6] SQ Le and O Gascuel. An improved general amino acid replacement matrix. *Mol Biol Evol*, 25:1307–1320, 2008.

[7] SV Muse and BS Gaut. A likelihood approach for comparing synonymous and nonsynonymous nucleotide substitution rates, with application to the chloroplast genome. *Mol Biol Evol*, 11:715–724, 1994.

[8] Cory L Strope, Stephen D Scott, and Etsuko N Moriyama. indel-Seq-Gen: a new protein family simulator incorporating domains, motifs, and indels. *Mol Biol Evol*, 24(3):640–649, 2007.

[9] S Tavare. Lines of descent and genealogical processes, and their applications in population genetics models. *Theor Popul Biol*, 26:119–164, 1984.

[10] Simon Whelan and Nick Goldman. A general empirical model of protein evolution derived from multiple protein families using a maximum likelihood approach. *Mol Biol Evol*, 18:691–699, 2001.

[11] Z. Yang. *Computational Molecular Evolution*. Oxford University Press, 2006.