Manual for COMPASS (The COMPletely Arbitrary Sequence Simulator)

Andrew Low

May 2017

Contents

1 Introduction

COMPASS is a BioPython-based simulator of sequence evolution along a tree using a rate matrix that can be completely defined by the user (standard rate matrices are, of course, also supported). It was designed to facilitate the study of coevolving residues, but could be used for other purposes as well, including simulation of sequences using standard rate matrices, codon models of evolution with non-standard genetic codes, evolution of phenotypic characters, and exploration of new models of sequence evolution.

2 Installation

Before installation, please note that you will need to have both BioPython and Numpy installed. To install, navigate to the COMPASS folder and type the following:

```
python setup.py build
python setup.py install
```

Assuming no error messages come up, COMPASS has successfully installed and can be used.

3 Quick Start

For an example, simply navigate to the folder which contains COMPASS as well as the included example parameter and tree files and type:

```
COMPASS.py -t examples/standard/example-tree.tre -p examples/standard/example-parameters.txt
```

This will create an alignment of 100 sites, evolving the sequence along the tree provided, according to the parameters in the example file, which in this case is an HKY matrix.

Within the examples folder there are 6 examples to help you get started. The first, in standard, is a simple simulation of nucleotides along the provided tree. The examples in partition, branch, and branch-site give examples of simulations that partition data into different models along sites, branches, and both sites and branches, respectively. The example in numeric gives an example of simulation of numeric characters, which could be useful for simulation of morphological characters. Finally, the example in mutation-selection shows the implementation of a mutation-selection model with varying population sizes (and therefore strength of selection) over different branches of the tree.

4 Input Formatting and Options

Most of the information needed by COMPASS is contained within the parameters file. This section will go over each subsection of the parameters file in detail, as well as the other pieces of information that COMPASS requires. Parameter files for many common matrices are included in the package, and can be used as templates. Other information needed is contained in a tree file which must be given to the program, as well as various command line options.

4.1 Tree File

COMPASS needs a tree file provided along which it will simulate evolution. The tree file must have branch lengths for all terminal and non-terminal branches included, and is expected to be in newick format. Should you wish to use a tree that is in a different format you are able to do so by using the –tree_format option (other than newick, available options are nexus, nexml, phyloxml, and cdao). An example of an acceptable input tree can be found in the example-tree.tre file included with this program.

4.2 Parameter File

The parameter file contains the majority of the information needed by COMPASS to simulate evolution. The three sections of the parameter file (@Frequencies, @Matrix, and @Multipliers), can be entered in any order, though it is recommended to keep them in the above order. Should you wish to add comments to your parameter file, this can be done by including the character # anywhere in the line you want to be a comment. Note that including this character anywhere in a line will make the entire line a comment, so be careful not to include any necessary information on comment lines. Blank lines will also be ignored.

It is possible to simulate with different models along different branches of the tree, in which case multiple parameter files will need to be specified. Full details on this can be found in section 6, Branch Models.

4.2.1 Frequencies

The first section of the parameters file should contain your desired nucleotide/codon/amino acid/phenotypic characters/whatever else you'd like frequencies. The format described here *must* be followed precisely. The example here uses nucleotides, but any symbols (except =,0, and *) can be used. Symbols can have as many characters as you wish and can be anything, except for the above noted =,0, and *. For example, the symbols 123, asdf, and 4r7yu7 would all be permitted. Only one entry per line is permitted. One important note is that the underlying BioPython code used to generate the sequence alignments won't allow sequences of different length which occur when symbols have different numbers of characters. If you want to get around this, setting the output format alignment to "phylip" will allow this.

@Frequencies
A=0.4
T=0.2
C=0.2
G=0.2

It is important to note that the first line of the frequencies section must be the header @Frequencies. Lines after that should be in the format Symbol=Frequency. No space should be left on either side of the equal sign, and the sum of all frequencies must be equal to 1, or the program will exit with an error.

The values for frequencies can be inserted into the rate matrix specified in the next section by inserting the symbol into the rate matrix (i.e. if you wanted a rate to be dependent upon the frequency of A, you would insert an A into the rate matrix).

4.2.2 Matrix

This section of the file must begin with @Matrix, followed by the actual matrix. An example matrix can be found below (in this case, a GTR matrix). The leftmost column represents whatever the current state is, and subsequent columns show the probability of moving from that state to another state. For example, if a site was currently in state G, the row that begins with G would be read to determine probabilities for each change. A change to state A is equal to the frequency of A, a change to state C is equal to the frequency of C multiplied by parameter a3, and a change to state T is equal to the frequency of T multiplied by parameter a5.

	A	C	G	T
A	0	C*a1	G	T*a2
C	A*a1	0	G*a3	T*a4
G	A	C*a3	0	T*a5
T	A*a2	C*a4	G*a5	0

Alternatively, one could specify a 16-state matrix in order to study nucleotide coevolution. For example, if one wanted to specify (exceedingly unrealistically) a matrix in which only substitutions between Watson-Crick base pairings occurred in proportion to the frequency of each Watson-Crick base pairing, it could be done with the matrix below.

	AA	AT	AC	AG	TA	TT	TC	TG	CA	CT	CC	CG	GA	GT	GC	GG
AA	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
AT	0	0	0	0	TA	0	0	0	0	0	0	CG	0	0	GC	0
AC	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
AG	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TA	0	AT	0	0	0	0	0	0	0	0	0	CG	0	0	TC	0
TT	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TC	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TG	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
CA	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
CT	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
CC	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
CG	0	AT	0	0	TA	0	0	0	0	0	0	0	0	0	GC	0
GA	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
GT	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
GC	0	AT	0	0	TA	0	0	0	0	0	0	CG	0	0	0	0
GG	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

The matrix specified in this section can use any symbols, and can have its rates represented as any combination of floating point numbers, frequencies, and multipliers. For example, the following matrix would be completely valid (as long as a1, a2, and and a3 are defined in the @Multipliers section).

	E	Q	SS	H
E	0	0.4	SS*H*a1*0.456	E
Q	Q	0	0.7	Q*a1*a2
SS	0	0	0	0
H	H	a2*a3*a1	1.2	0

Formatting notes: Entries in the matrix should be tab-separated (in practice, any whitespace will work, but this creates exceedingly ugly-looking matrices). The first line of the matrix must contain all the symbols that you wish to have as part of the evolutionary process, and every subsequent line must begin with a symbol and be followed by the transition parameters. As in the above example, the order of symbols in the first row must also be the order of symbols in the rows below (i.e. If your first row starts with an E, the first row containing parameters must also begin with an E, the second symbol in the first row is a Q, and so the second row containing parameters begins with Q, etc.). Transition parameters can be any combination of frequencies, multipliers, and floating point numbers, separated by a "*". No spaces should be left on either side of the "*" symbol, as COMPASS tries to parse your matrix based on whitespace.

4.2.3 Multipliers

The user can designate any number of multipliers they wish to use in the rate matrix in this section. As with previous sections, this section begins with @Multipliers. Subsequent lines should be in this format: Name_of_Multiplier=Floating_Point_Value.
As with the frequencies section, the multiplier can be named anything (except for =,0, or *), and there should be no space inserted on either side of the equal sign. Again, only one entry per line is permitted. An example of a multipliers section can be found below.
@Multipliers
a1=0.3
b=0.4
w=1.32
Symbols that have already been used in the @Frequencies section cannot be used in the @Multipliers section, and vice versa, or the program will exit with an error message.

5 Command Line Options

Full descriptions of all command line options can be found by reading this section, or typing the following into the command line:

```
COMPASS.py --help
```

5.1 Mandatory Arguments

5.1.1 Tree File (-t)

The first positional argument given must be a file containing the phylogenetic tree along which you wish to simulate sequence evolution. This tree can be rooted or unrooted and in any of the following formats: newick, nexus, nexml, phyloxml, cdao. Note that the program expects a newick tree by default, and using any other format will require the use of an optional argument (see optional argument –tree_format).

5.1.2 Parameter File (-p)

The second argument provided must be a parameter file, which is described in detail in the above section. This can either be a file entirely created by you, or one of the premade files included with the program, although you will likely wish to modify the parameters in whatever matrix you choose in order to better model the situation you are attempting to simulate evolution for.

5.2 Optional Arguments

5.2.1 –tree_format (-tf)

This argument allows you to specify a non-newick format for your input tree. Allowed types are those supported by the Phylo module of BioPython: newick, nexus, and phyloxml.

5.2.2 –number_sites (-ns)

This option specifies the number of sites you wish to have in your alignment. The default setting is to have an alignment of 100 sites. It is important to note that if the symbols in your alignment contain more than 1 character (i.e. you are modelling RNA coevolution and using a 16-state matrix with AA, AU, AC, AG, etc., as your states) you would end up with a sequence alignment of length 200. If you wished to have only 100 sites in your alignment in this example, you would set this parameter to 50.

5.2.3 –gamma_categories (-gcat)

The number of discrete gamma categories you wish to use. The default for this parameter is 4, and it will be used unless the continuous gamma option is specified (i.e. if you do not enter this option, by default COMPASS will use a discrete gamma distribution with 4 categories). Rate variation among sites will always follow either this option or a continuous gamma distribution. Should you wish to have no rate variation among sites, set this parameter equal to 1.

5.2.4 –cont_gamma (-cg)

Enabling this parameter will force usage of a continuous gamma distribution for the distribution of rates across sites. This will make use of the alpha parameter (see next section), so if you wish to use a different alpha parameter than the one provided, be sure to specify it.

If both this option and a discrete number of gamma categories are specified, this option will take precedence.

5.2.5 –alpha (-a)

The alpha parameter to be used to create your gamma distribution. By default, this is set to be 1.0. The higher the number you set this parameter to, the more homogenous rate variation among sites will be. The value for this argument must be positive, or the program will exit with an error message.

5.2.6 –invariant_sites (-is)

This parameter will cause a portion of your sites to be invariant, with the default portion of invariant sites being set to 0. The argument for this parameter is a floating point number between 0 and 1, with 0 being no invariant sites, and 1 being all invariant sites. Of note, this parameter only specifies the probability that sites will be invariant (i.e. specifying 0.1 means that each site has a 10 percent chance of being invariant), so the number of invariant sites that occur in your alignment will likely not perfectly reflect the percentage of invariant sites specified, particularly for short alignments. The longer the alignment, the better the agreement between specified and actual number of sites.

5.2.7 –output_format (-of)

The output format you wish your sequence alignment to be created in. The default is fasta, while other available options are clustal, fasta, phylip, phylip-sequential, phylip-relaxed, and stockholm. Having your characters output as tab-separated values is also an option, using tsv. Note that these names are case-sensitive.

5.2.8 -m (Create Substitution Mappings)

The -m option will create a mapping of where along the specified tree each mutation occurred, with one file created for each site in the alignment.
These mappings are written in a format similar to the newick format. An example follows for a simple 3-taxa tree:

```
((Species3\_T:0.2:T:0.5:G:0.4:A,Species2\_A:0.2:A)A:0.8:A,Species1\_A:1.0:A)A;
```

In this mapping, the ancestral state was A, and remains A at all terminal nodes except for species 3, which had a substitution from A to G occur at 0.4 after its split with species 2, followed by a G to T substitution 0.5 after the initial substitution. There is then no change for 0.2 until the observed state. This can be visualized on the following tree:

```
|----------(A) Species1
|(A)
|        |--(A) Species2
|--------|
         |----/-----/--(T) Species3
            A->G  G->T
```

5.2.9 -g (Create Gamma Rate Information File)

The -g option will create a file which contains details on which rate (multiplier of branch length selected from the gamma distribution) was selected for each site, as well as the number of events that occurred at each site along the length of the tree.

5.2.10 –number_replicates (-nr)

The number of times you wish to run your simulation. By default only one replicate is created, but you can create as many as you like. Output files will be numbered starting from 1, and going to whatever number is specified here.

5.2.11 –output_name (-o)

By default, sequences you create will be named simulated(replicate_number).file_extension. If you wish to have a different base name for your output files, use this option to specify whatever other name you want to use.

6 Partitioning Data

Using COMPASS, one can also partition data so that part of a sequence will evolve under one set of conditions while other parts of the sequence will evolve under other conditions. This option can also be used to incorporate heterogeneity with respect to branch length for different sites along your sequence. In order to partition data, the wrapper script *wrap_COMPASS.py* should be used. This script takes a control file as an argument, along with a few command line options, which are explained in subsequent sections.

6.1 Control File

The control file for *wrap_COMPASS.py* serves the function of accepting the arguments necessary to run COMPASS for each partition. There is no limit on the number of partitions that can be included.
To use this program, type:

```
wrap_COMPASS.py name_of_control_file
```

There are 8 columns that must be included in this file, and they must be in the same order in which they appear in the file *example_partition.txt*. The 8 columns are:

Matrix
Specifies the file that contains the Matrix, Frequencies, and Multipliers for the partition. If this file resides in a directory other than the one that COMPASS is running in, the full path to the file must be specified. If running COMPASS with several models on different branches, the parameter files should be separated by a comma, and in the same order they would be in outside of partition mode.

Tree
The tree file that is to be used for this partition. Trees for each partition should all have the same topology and number of species, though branch lengths may vary. By default this is expected to be a newick tree, but other formats are available with the –tree_format option. Trees for each partition must be in the same format, and as with the *Matrix* section, if the tree files are not in the same directory as COMPASS, the full path must be specified.

Number_Sites
The number of sites to be included in the partition.

Gamma_Dist
This column controls the gamma distribution used for rate heterogeneity among sites. Should the value for this column be set to 0, a continuous gamma distribution will be used. Setting this column to any positive integer will make COMPASS use a discrete gamma distribution with that number of rate categories.

Alpha
The alpha parameter for use in the gamma distribution. Can be any value between 0 and positive infinity.

Invariant
The proportion of invariant sites to include within this partition. Must be between 0 and 1.

Gamma_Info
Setting this parameter to 1 will enable information about the gamma multipliers to be used for each site as described previously with the -g option. One mapping file will be created for each partition. Setting this

parameter to 0 will turn this option off.

Mapping
Setting this parameter to 1 will enable the mappings for each site to be recorded as previously described with the -m option. Setting this parameter to 0 will disable this option.

6.2 Command Line Options

The command line options for *wrap_COMPASS.py* are very similar to those for *COMPASS.py*, and are described below.

6.2.1 –tree_format (-tf)

The format that input trees are in. By default this option is set to newick, but it can be changed to nexus or phyloxml. Trees for all partitions must be in the same format.

6.2.2 –output_name (-o)

The base name for your output files. By default it is set to "simulated".

6.2.3 –number_replicates (-nr)

The number of replicates to be made of the sequence. The first will be named base_name1.format, the second base_name2.format, and so on. By default, this is set to be 1.

6.2.4 –output_format (-of)

The output format for your sequences. By default the output format is fasta, but this can also be set to phylip-relaxed, clustal, phylip, phylip-sequential, and stockholm.

7 Branch Models

Simulating with different models along different branches of a tree is also possible with COMPASS. The process to set this up is relatively simple, and can be combined with the data partitioning from the previous section to simulate alignments models that vary both among sites and branches.

An example can be found in the examples folder, in the branch directory, and an example that combines both branch and site models is found in examples/branch-site.

7.1 Tree Setup

The tree used to simulate evolution must have appropriate markers for the different models of evolution to be used. The marker chosen to do this in COMPASS is the "$" symbol, followed by the model number. This marker must be placed at the end of the species name, and every branch must be marked (i.e. if an ancestor is marked, children of that ancestor will not also be marked by default, they must also be marked).

Consider the following example, in which a tree has four species: human, chimp, gorilla, and gibbon. The user wishes to have three different models of evolution used to simulate this data: one for human, one for chimp, and one for the rest of the tree. To do this, leave unmarked all branches except for the human and chimp branches. On the human branch, change the species name to human$1, and change the chimp branch to chimp$2.

Note that the markers *must* always be placed at the end of the species name, and must start at 1 (i.e. in a tree with two models, one marked and one unmarked, $1 is the only marker that can be used - $3 would not be allowed). Any number of models can be used, as long as a parameter file is specified for each marker and the marker numbers go up sequentially ($1 and $2 for three total models, $1, $2 and $3 for four total models, etc.)

7.2 Parameter Files

If you are using different models along different branches, one parameter file must be specified for each of the different models. To do this, multiple parameter files should be specified after the -p flag, each separated by a space. The order of these parameter files is important. The first parameter file specified is always used for unmarked branches, the second for any branch marked with $1, the third for branches marked with $2, and so on. In the above example, the command typed would be something like:

```
python COMPASS.py -t marked_tree.tre -p first_model.txt human_model.txt chimp_model.txt
```

8 Limitations and Known Issues With COMPASS

While generally quite versatile and reliable, COMPASS does have some limitations and known issues.

- Version of Python: While in principle any version of Python 2.7.x should work, some issues have been found with older versions. Use of Python 2.7.11 or newer is recommended.
- Size of tree: In testing on a machine running MacOSX with 8 GB of RAM trees with up to 50,000 tips have been simulated, but there have been reports of issues on trees of 10,000 tips. This may be machine dependent, as there should be no upper limit.
- Number of sites in sequence: Tests with up to 10 million sites in the alignment have been carried out, and are able to run (albeit slowly - parallelization and further optimization in the future hope to speed up simulations), but simulating gigantic (i.e. trillions or more) alignments can cause crashes.
- Tree creation: COMPASS does not currently support simulation of phylogenetic trees, although this a potential area for further development. In the meantime, TreeSim implements many useful models for simulation of trees (https://CRAN.R-project.org/package=TreeSim).