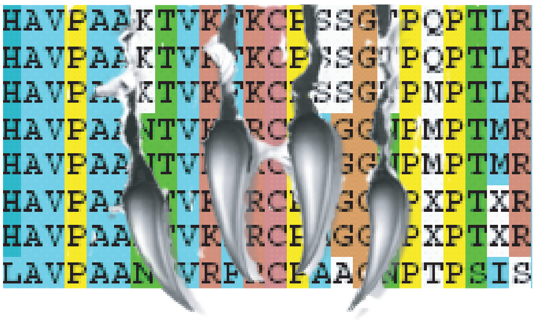

TIGER Manual

Tree Independent Generation of Evolutionary Rates

Carla A. Cummins and James O. McInerney

Table of Contents

Introduction

TIGER is a piece of open source software for identifying rapidly evolving sites (column in an alignment) in alignments of all types of data (molecular, morphological etc.). Sites like these are important to identify as they are very often removed to improve phylogenetic reconstruction. When a site is changing very quickly between species it no longer holds any phylogenetic information and therefore is simply a source of noise. Use of TIGER can (a) allow you to see the amount of rapid evolution and noise in your alignment and (b) provide a quick and easy way to remove as many of the "noisy" sites as possible.

TIGER uses disagreement between site patterns as a proxy for rapid evolution; that is, a site that does not disagree with other sites in the alignment is generally a very slowly evolving or constant site. A site with lots of disagreement is considered rapidly evolving (Cummins & McInerney, in prep). TIGER assesses the disagreement and categorizes the sites based on the rates. In this software the categories are called bins and are user definable (see options, page 6). Bin1 will contain the constant sites and the highest bin will contain the most rapidly evolving sites.

TIGER outputs a NEXUS formatted file. In the PAUP block the bins of sites are defined as character sets (Charsets). This allows an easy means for removing the bins (see example, page 9). For cases where PAUP cannot be used an additional Python script is distributed for removing the character set required (see information page 10).

TIGER is implemented in Python therefore it is multi-platformed.

System Requirements

- On UNIX systems, a working version of Python 2.5 or 2.6. There is poor compatibility between Python releases. TIGER *may* work on other versions of Python, but these versions have been tested, so if it does not work with your current version, please resort to one of the aforementioned versions.
- The Windows distribution comes with everything needed, so has no further requirements.

Installation

Unix (Mac & Linux)

The files "tiger" and "tiger_fns_102.py" are required for TIGER to run. To access the necessary files throughout your filesystem, place them in your local binary directory (usually located at `/usr/local/bin/`). This avoids unneeded replicates of the scripts. Once the files are accessible system-wide, proceed to use TIGER as directed in the rest of the manual.

Windows

The Windows distribution consists of a folder named dist/. Add the path to this folder to your Windows environment variable PATH to access TIGER system-wide. For information about this see: http://msdn.microsoft.com/en-us/library/ee537574.aspx. Once the folder is on your path, proceed to use TIGER as instructed in the rest of the manual.

Using TIGER

Input

TIGER accepts input in FastA format. If your data is not in this format, readseq (http://www-bimas.cit.nih.gov/molbio/readseq/) can convert almost all sequence data formats to FastA. **NOTE**: all characters, except those defined as ambiguous (see -u option) are treated as a separate character state. This includes white space. Remove non-sequence characters from your dataset prior to TIGER analysis.

Output

TIGER output is in NEXUS format. This allows us to define *Charsets*, which is a command that can be understood by many NEXUS compatible softwares (PAUP, MrBayes, Mesquite). The sites that fall into each bin (bin info, page 7) are defined as a charset. This allows the user to customize the data as required. NOTE: TIGER does not remove any data. It provides the user a simple means for subsequent data removal if desired. Commands such as include and exclude allows the user to quickly remove an entire category of sites.

Running TIGER – a simple example

Once TIGER is installed, you simply use:

```
tiger –in input.fas
```

Assuming `input.fas` is the file you wish to run it on. As a default TIGER prints its output to the screen. To save the output to a file you will need to redirect it to your file using the ">". For example:

```
tiger –in input.fas > output.nex
```

Assuming `output.nex` is the file you wish to write to.

TIGER options

-h : Quick Help

There is a condensed manual available by using the -h option:

```
tiger –h
```

This lists the options available using the software along with a simple example.

-in: Input File

As shown in the simple example above, -in is used to define the input file. Remember that TIGER accepts FastA format.

-b : Number of Bins

As mentioned in the introduction, TIGER scores sites based on the amount of agreement it displays with other sites in the alignment (Cummins and McInerney, in press). When all sites are scored TIGER splits the range of values into equal partitions, each called a “bin”. Sites are placed into their correct bin based on which partition range in which its score falls. The number of bins, or partitions, is set to 10 as a default, but can be changed using the -b option. For example, to run an analysis using 20 bins you would use:

```
tiger –in input.fas –b 20
```

-f : Output Format

By default, TIGER's output is in NEXUS format with the bin number placed under each site directly into the file. This allows easy viewing of the contents of each bin. If you wish to see the scores of the sites instead of the bin you may use the -f option. To see the rates that are assigned to the sites rather than their bins use:

```
tiger –in input.fas –f r
```

"r" standing for "rates". There is also the option of "sorting" your alignment based on its rate. This means that the columns of the alignment will be rearranged from the slowest evolving sites to the fastest in order. This allows easy comparisons of bin contents. To have your output sorted with the bin numbers displayed use:

```
tiger –in input.fas –f s
```

"s" standing for "sort". If you wish both options to be on (sorted with rates displayed) use:

```
tiger –in input.fas –f r,s
```

(Note: no space between the "," and the "s".)

-rl : Rate List Output

A list of rates at each site may be optionally written to a specified file. To use this feature:

```
tiger –in input.fas –rl ratefile.txt
```

This will write the list to the file "ratefile.txt". Each number corresponds to a column (in the same order as they occurred in the input file).

-ptp : PTP Test

The PTP test implemented in TIGER identifies sites that are significantly disagreeing. To do this we randomize the characters at the site in question a number of times (see -z option below) and calculate its disagreement score. We take note of the proportion of times this random sites score is less than the original sites score ie: the raw site shows more disagreement with the data than the random site. This is the p-value. If the p-value is greater than the designated cutoff value (see -pv option below) the site is assumed to be significantly disagreeing.

By default TIGER **does not** execute a PTP test (quicker execution...). Use -ptp to switch the PTP test on. The results of the test will be displayed as another character set in the PAUP block: Charset Sig_Disagreement (for Charset info, see page 9). To run use:

```
tiger –in input.fas –ptp
```

-z : Randomizations

Customize the number of randomizations to perform using the PTP test. Default is 100. To change to 500, for example, use:

```
tiger –in input.fas –ptp –z 500
```

-p : P-Value

You may change the p-value cutoff to be used in defining significance in the PTP test. The default is 0.05. To change the p-value to 0.1 use:

```
tiger –in input.fas –ptp –pv 0.1
```

-pl: P-Value List

Like the -rl option, a full list of p-values may be written to a specified file. To save the list to a file named “pvalues.txt” use:

```
tiger –in input.fas –ptp –pl pvalues.txt
```

Note that a list of p-values can only be written when the PTP test is selected using the -ptp option.

-u: Uncertain Characters

Some characters in an alignment can denote uncertainty or missing data. As a default *?* is considered the only uncertain character, but this may be changed using the -u option. To define a different character, such as *x*, use:

```
tiger –in input.fas –u x
```

To define multiple characters use:

```
tiger –in input.fas –u x,?,!
```

Separate characters using only a “,” with no space.

-v: Version

This simply displays the version of the software you are using.

Version & Bug Information

TIGER is a relatively new piece of software and is on version 1.02.

It has been extensively tested, but with all software there can be no “bug-free guarantee”. If a bug is encountered, you have a question regarding the software or you simply would like to see a useful new feature added please contact: carlacummins@gmail.com.