



Manual

Introduction	Introduction	Basics	Files	Objects	Characters
	Why?		Menus		Taxa
	Publication		Windows		Trees
	Support		Charts		Glossary
	Credits		Scripts/Macros		
	Help		Modules		
	Web Site		How		

Analyses

[Character Evolution](#)

[Simulations &](#)

[Randomizations](#)

[Population Genetics](#)

[Molecular Data](#)

[Continuous Characters](#)

[Studies](#)

Mesquite: A modular system for evolutionary analysis, version 1.0

Wayne P. Maddison, University of British Columbia
David R. Maddison, University of Arizona

This manual introduces you to the Mesquite system for analysis in evolutionary biology. It explains the basic operation of the system (the links above) and some standard analyses (the links at left).

Mesquite's analyses currently include phylogenetic analyses (parsimony, likelihood, comparative method, simulations and randomizations of characters and trees) and population genetics analyses (coalescence). You can also use Mesquite as an editor for phylogenetic data files. Mesquite's web site contains an outline of [Mesquite's features](#).

Because many analyses are possible and those available depend on what modules are installed and loaded, a comprehensive manual cannot easily be written. For this reason, many analyses available in Mesquite are not included in this manual. You are encouraged to explore, and invent new analyses.

Mesquite's web site is [here](#); check it for updates.

Getting started

To begin using Mesquite on the Mac OS, Unix/Linux or Windows, see the [installation instructions](#). We encourage you to explore the example files in the "examples" directory of the Mesquite_Folder.

Reporting Bugs

We encourage you to report bugs or misbehaviour of Mesquite. You may find a problem that seems so obvious to you that you expect we must have seen it. Please report it anyway, because perhaps it occurs only in particular circumstances that we didn't test, or it occurs with a combination of options that we haven't tested recently. The [Support](#) page has more details on reporting bugs.

Conventions of this manual

For compactness, we will use a special convention to refer to menu items. For example, [File>Save File](#) refers to the Save File menu item of the File menu. ([Tree Window](#))[Drawing>Tree Form>Diagonal Tree](#) refers to the Diagonal Tree menu item of the Tree Form submenu of the Drawing menu that is associated with the Tree Window. By referring to a menu as "associated with" a window, we mean that that menu is present in the menu bar at the top of the screen when the window is frontmost (on the MacOS), or that the menu is embedded within the window (on the Windows OS and most other operating systems).

Information for Developers

Mesquite is modular, and can be extended by adding modules into the folder "mesquite" in the folder "Mesquite_Folder". Developers interested in writing modules should contact us at dev_elp@mesquiteproject.org. Information on development for Mesquite is also available [here](#), although it is out of date.

Copyright © 2002-2003 by [Wayne P. Maddison](#) and [David R. Maddison](#).
All rights reserved.

Mesquite installation for MacOS

(Please email us (info@mesquiteproject.org) with questions or comments about downloading Mesquite).

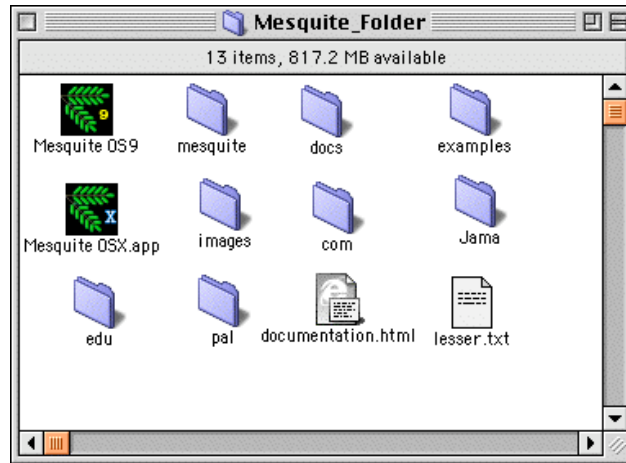
Download one of these, which contain Mesquite for use on the Mac OS:

- [.dmg \(Disk Image\) file \(this may not work on versions of the Mac OS prior to OS X\)](#)
- [.sit \(Stuffit\) file](#)

For Macintosh OS 8 or 9

Requirements: MRJ 2.2 (www.apple.com/java)

Instructions: Download and unstuff the .sit file above. This will create a folder approximately as follows:



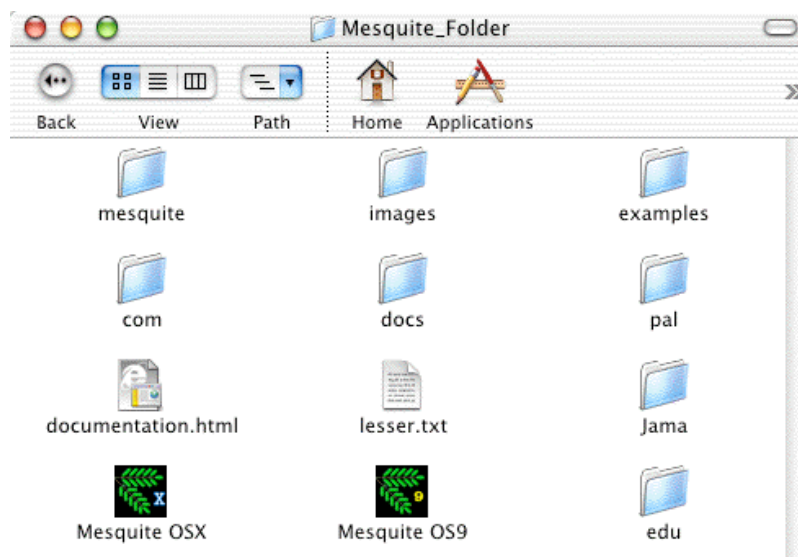
To start Mesquite, you can double click either the icon "Mesquite OS 9", or double click a data file in "example files".


NOTE: If you have a **previous version** of Mesquite installed in your system, we advise that you delete it before running the new version.

For Mac OS X

Requirements: OS X 10.2 recommended; 10.1 with java update 1 has some especially frustrating bugs; Mesquite is more or less unusable with OS X 10.0. Mesquite at present is not designed to run on java 1.4.1 on Mac OS X, and instead will use java 1.3.1 if available.

Instructions: If you download the dmg file above, double click on the file to mount the disk image. Drag the contained "Mesquite_Folder" to your hard drive to install. Otherwise, download and unstuff the .sit file above. Both methods will create a folder "Mesquite_Folder" whose contents will be approximately as follows:





To start Mesquite, you can double click either the icon "Mesquite OSX", or double click a data file in "examples".

NOTE: If you have a **previous version** of Mesquite installed in your system, we advise that you delete it before running the new version.

More details of issues of using Mesquite under Mac OS X are described in the [Support](#) page.

Mesquite installation for UNIX/LINUX

(Please email us at info@mesquiteproject.org) with questions or comments about downloading Mesquite).

Requirements: JRE or JDK 1.1.8 or better (java.sun.com). Recommended: 1.4.1 or higher. (It is best to get Sun's Java VM; Mesquite does not run on some third-party virtual machines.) See notes on the [support](#) page, especially regarding issues with window and dialog size and placement.

- [.tgz file](#)

Instructions: Download and decompress one of the files above. It will create a directory called "Mesquite_Folder" in which all the relevant files reside. The main class file is `mesquite.Mesquite`. To start Mesquite you can use `jre` or `java`. For instance, on a Linux box we've used, the following command is sufficient:

```
jre -cp /home/myuser/Mesquite_Folder mesquite.Mesquite
```

where `/home/myuser/Mesquite_Folder` could be replaced by whatever is the path to the Mesquite_Folder. On some systems the java virtual machine is started by "java" instead of "jre", and thus the command would be:

```
java -cp /home/myuser/Mesquite_Folder mesquite.Mesquite
```

Depending on your configuration, you may need to give the explicit path to the java virtual machine, as in:

```
/usr/java/jre1.4.2/bin/java -cp /usr/local/Mesquite_Folder mesquite.Mesquite
```

If you've changed the name of Mesquite_Folder to, for instance, "Mesquite1", you may need to use a command like:

```
/usr/java/jre1.4.2/bin/java -cp /usr/local/Mesquite1 mesquite.Mesquite
```

To make it easier to start up Mesquite each time, you may want to make a shell script containing the appropriate command.

Once Mesquite has finished loading, go to the File menu to open a file.

If the current user directory is not the Mesquite_Folder, and Mesquite has not been run before, Mesquite may ask you to find the file "manual.html" which resides in the **Mesquite_Folder/docs/mesquite/** directory. This will help Mesquite find and remember where its files are.

Window Managers

Mesquite attempts to place windows in particular places on the screen for ease of use, using standard Java calls. Some window managers override this, resulting in haphazard placement of windows. You may need to change your window manager if this sort of thing is happening.

More details of issues of using Mesquite under Linux/UNIX are described in the [Support](#) page.

Mesquite installation for Windows

(Please email us (info@mesquiteproject.org) with questions or comments about downloading Mesquite).

NOTE: If you previously installed an older version of Mesquite, we recommend you delete the **Mesquite_Folder/mesquite** directory before installing the new version.

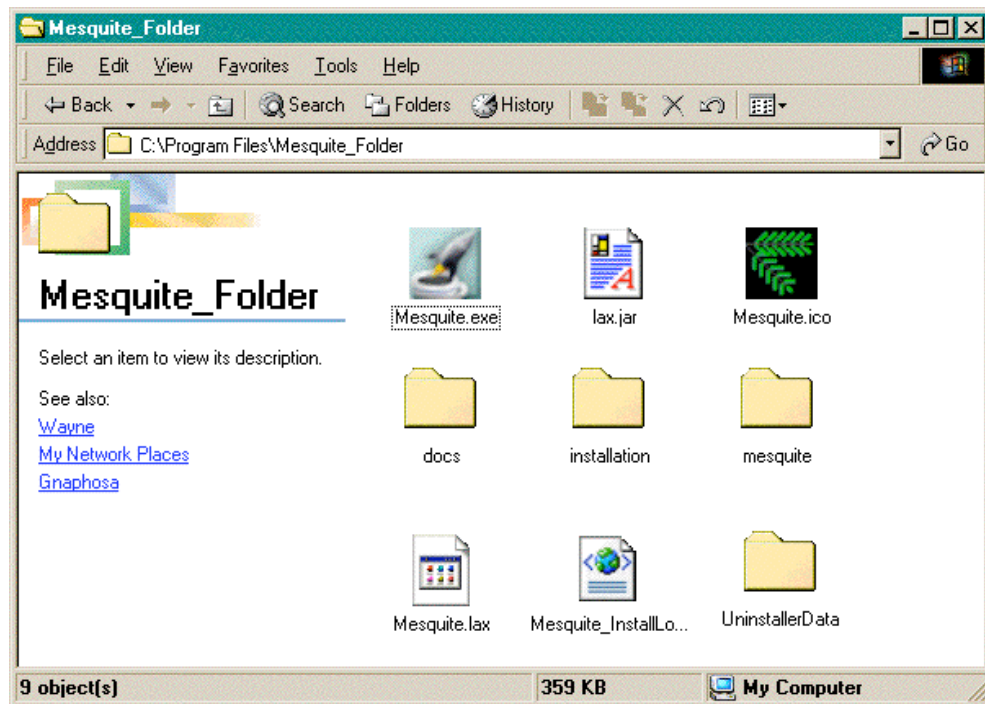
Requirements: JRE or JDK 1.1.8 or above; 1.4 or above recommended. (java.sun.com)

Instructions: Installation is a two step processes:

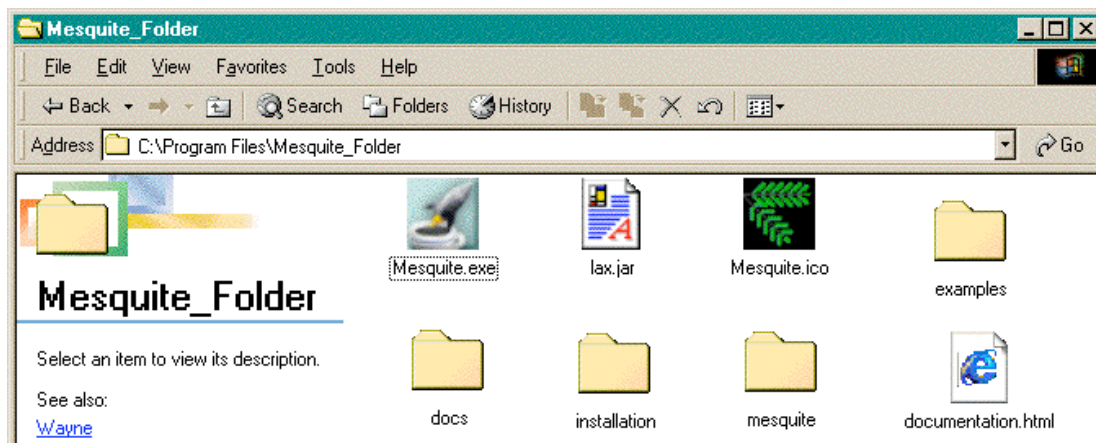
First, download and run the installer "[mesquiteInstall.exe](#)" (this installer was made using InstallAnywhere from www.zerog.com).

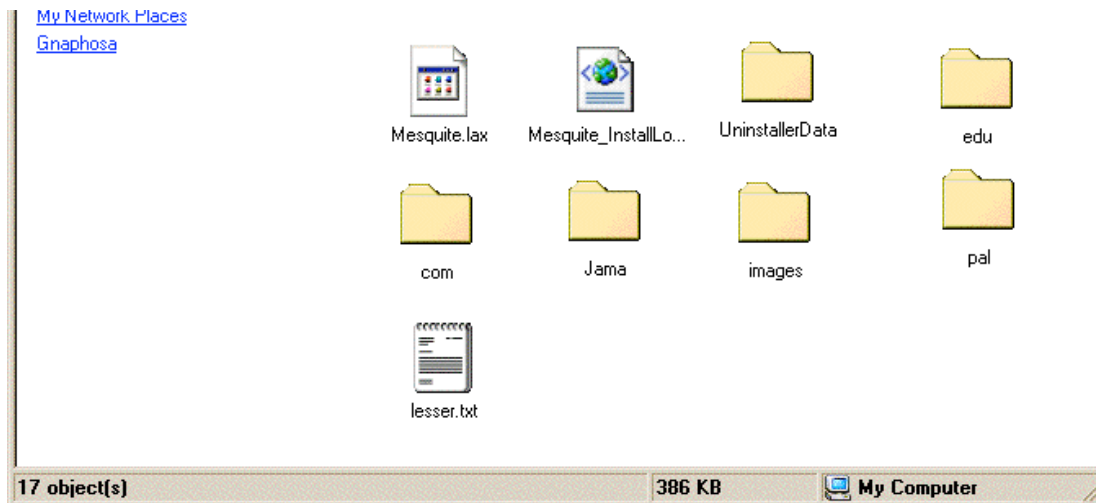
When you run the installer, it may ask you to select a Java Virtual Machine. It's best to choose one made by Sun, which will be in a directory named JRE or JDK (e.g., c:\Program Files\JavaSoft\JRE\1.3.1\bin\java.exe). It is also best to choose a more recent one (e.g., choose 1.4 over 1.3.1 over 1.2.2 over 1.1.8).

This should create the directory "**Mesquite_Folder**", and within it, the directory "**mesquite**" and several files, approximately as follows (the icons and details may differ).



Second, download and unzip the archive "[mesquite.zip](#)". It will contain one directory called "**put contents in Mesquite Folder**", which contains at least four other directories, "**mesquite**", "**com**", "**images**" and "**example files**". Place all of the contained directories within Mesquite Folder, so as to replace "mesquite" and add the other directories, to yield something like the following:





Mesquite can then be started up by starting up the executable "Mesquite". The first time Mesquite runs, a welcome dialog box will appear. Once Mesquite has finished loading, go to the File menu to open a file.

In the first step of the installation processes, the Mesquite package itself is not installed, just files that help Mesquite run. The Mesquite package is within the mesquite.zip archive. The reason we are burdening you with a two-step installation process is that, when future versions of Mesquite are posted online, you won't have to run the installer again (and we won't have to create new installers). You'll only have to download a new Mesquite system and replace the "mesquite" directory and the other directories.

If you had previously installed an older version of Mesquite, please delete the old Mesquite_Folder/mesquite/ directory before replacing it with the new one. If you merely copy the new mesquite/ into the location of the old, and let the system overwrite the contents, some old modules will remain because of the counterintuitive behavior of Windows in replacing directories.

If you are familiar with Java under Windows and prefer to start up programs from the DOS prompt, you can run Mesquite without having done the first step of the installation process, as long as your class paths to the virtual machine and so forth are prepared. Go to the DOS prompt and type:

```
jre -cp "<path to Mesquite Folder>\Mesquite_Folder" mesquite.Mesquite
```

or

```
java -cp "<path to Mesquite Folder>\Mesquite_Folder" mesquite.Mesquite
```

where <path to Mesquite Folder> might be something like C:\Program Files

If the current user directory is not "Mesquite_Folder", and Mesquite has not been run before, Mesquite will ask you to find the file "manual.html" which resides in the **Mesquite Folder\docs\mesquite** directory. This will help Mesquite find and remember where its files are.

Misbehavior of Mesquite under Windows

Occasionally the menus of Mesquite will show incorrect headings (this is apparently due to a bug in the Java Virtual machine, not due to a bug in Mesquite). For this reason there is a Reset Menus item in the File menu; it should help restore the correct menus.

If you set Mesquite to be the default application to open NEXUS files (with the .nex extension), it should be able to open files given to it. However, under some circumstances it may claim the file was not found. Please report this.

More details of issues of using Mesquite under Windows are described in the [Support](#) page.

Why Mesquite was made

We give two answers, the [practical](#) and the [poetic](#), and a comment on the relationship between [MacClade](#) and [Mesquite](#).

The practical answer

Mesquite represents a new approach to computing for evolutionary biology. In recent years there has been a proliferation of computer programs for phylogenetic analysis, each designed for some particular analysis (e.g., see [Felsenstein's compilation](#) of programs). As these often involve unique file formats and user interfaces, it is difficult for users to move from one to another. Users tend to become constrained to a few familiar analyses, since any given program can't do everything, and each program has costs in learning. As a programmer one would like to respond by making a program that does everything, but there are now too many analyses available or conceivable for a single programmer or programming team to keep up. We have seen the impact of these constraints with MacClade: some users perform particular analyses in MacClade not because they are the most appropriate analyses for their questions, but simply because they are available in a familiar program. We would like to add more flexibility to MacClade, but in a monolithic program this can be difficult to do, and even if easy, there are more proposed methods than we could maintain in MacClade.

Hence, our goal was to design a general system for phylogenetic computing to which different programmers could contribute modules. Bringing different analytical tools into a common system increases possible analyses more than additively. In the end, the system has grown beyond being strictly phylogenetic, including capabilities for calculations involving characteristics of many organisms (e.g. population genetics and morphometrics) that need not involve phylogeny.

A second goal of Mesquite is to provide a graphical user interface that will operate, more or less without modification, under different operating systems (being written in Java).

Modularity and Flexibility

"Modularity" in computer programming might follow different models. It could follow the "[Mr. Potato Head](#)" model, in which there is a central program to which different peripheral calculations can be attached in specific places. This allows useful, but limited, flexibility. Or, modularity could follow the "[Lego](#)" model, in which building blocks are attached to other building blocks, and so on indefinitely. This allows nearly unlimited flexibility. Mesquite's modularity is somewhat of a hybrid between these: there is a (small) central starting point to which modules attach, but from there modules can be attached to modules attached to modules, indefinitely, leading to considerable flexibility in the analyses that can be constructed.

To give an idea of the flexibility, consider the calculation of the parsimony score of a tree, the treelength. A treelength calculating module takes as input information a tree, and responds by returning its length. Such a module belongs to the general class of modules that return a number when passed a tree. Other modules belonging to this class ("NumberForTree") could return the likelihood of the tree, or a measure of the asymmetry of the tree's branching, or a measure of the tree's discordance with a containing species tree. A Tree Legend module can be written (and has been) that displays the treelength in a legend in the tree window, but the Legend module is designed so that the user can choose to display any other number for the tree, such as its likelihood, asymmetry, or discordance. If a programmer creates a new module to calculate a number for a tree such as the longest branch-length path from root to tip, and a user installs the module, then the longest path measurement would automatically become another option for the tree legend.

The Tree Legend is not the only place where analyses could use numbers for trees. A charting module could display the numbers calculated for a whole series of trees, or a tree search module could use the numbers to find a tree with minimum or maximum values for the number. When such modules are made, they can automatically have access to whatever NumberForTree modules are available. Thus, the chart could show treelength, or likelihood, or asymmetry, or discordance, or longest path. Likewise, the tree search module could seek to optimize any of those. If a programmer makes a new module to analyze numbers for trees, then suddenly all existing NumberForTree modules have a new context in which they can be analyzed. If a new NumberForTree module is made, it will appear as a new option under each of the modules making use of NumberForTree. Hence the number of alternative analyses rises as the product of numbers of modules of different interacting types.

Of course, the trees used had to come from somewhere. One module might supply the trees stored in a file, another might simulate trees using a simple markovian model of speciation and extinction, another might simulate trees as gene trees coalescing within a species tree. Characters likewise might come from a stored matrix, or might be simulated by a stochastic module of evolution, or might represent reshufflings of existing characters. This means that any calculations using trees or characters can either do their calculations on observed data and reconstructed trees, or can derive null distributions under stochastic models. The calculations don't have to do anything special to achieve this flexibility; they simply let the user choose the sources of trees and characters.

(For more details about modularity, see [How Mesquite works](#))

A community of programmers

Our hope is that building-block style of the Mesquite system will encourage programmers to write modules for their own favorite analyses. Another attraction of the Mesquite system is that many of the details of reading and writing of files, user interface and graphical display are already taken care of, and the programmer might worry only about a single calculation. The system is built in Java and is therefore platform independent. It is also possible for programmers to link in code written in C, C++, or some other language.

We have attempted to design the system so that a programmer's efforts can be recognized as an independent, citable contribution. Modules or suites of modules can have their own names, own manuals, be distributed and cited separately. They simply run within the Mesquite system.

Mesquite source code is available for [download](#). This allows other programmers to modify existing source to create new modules.

The poetic answer

The goals of Mesquite are these:

To change the economics of imagination in evolutionary biology – There are three ways we envision Mesquite stimulating imaginative ideas and their successful spread:

- **Stimulating the creation of ideas: analyses.** With multiple alternative modules available for various parts of an analysis, and with modules specializing in questions from various branches of evolutionary biology (e.g., phylogenetics, molecular evolution, population genetics, geometric morphometrics) the diversity and scope of analyses that can be constructed by combining different modules is great. Individual users can carry their imaginations through to an analysis that no one has tried previously. Indeed, Mesquite, by offering the alternatives to be combined, doesn't merely provide analytical tools for questions that have existed: it suggests and provokes new questions.
- **Stimulating the creation of ideas: biology.** As does [MacClade](#), Mesquite has an emphasis on visualization and exploration. An idea – whether a particular hypothesis about the evolutionary history of a group, or a stochastic model of a process – can be followed through to its consequences, and visualized. A biologist can ask "What if this were the phylogenetic tree?" and a character's evolution can be reconstructed or simulated on this tree, and the results visualized. A biologist can ask "What if the population had population sizes fluctuating in this way?", and coalescence can be simulated, and the results visualized. In providing users with the tool to ask "What if?" questions, Mesquite provides an extension of the imagination. Such tools are vital in a field whose ideas have consequences that are difficult to predict or grasp without the aid of a computer.
- **Enhancing the efficient distribution of ideas: programs.** The imagination of theoreticians and programmers has produced many valuable ideas for approaches and methods, and many valuable programs to implement them. However, some of the ideas haven't been translated to programs, and many of the programs haven't been as much explored and used as would have been good. We don't know, as a field, how many important ideas will lie unused for decades until they are rediscovered. By allowing the programmer to focus on the precise idea proposed (Mesquite providing much of the housekeeping code for the programmer), Mesquite may allow some ideas, that might never have been implemented, to be realized as tools. By providing a fairly user-friendly context in which modules can operate, Mesquite may encourage some programs to be used more broadly and more easily than otherwise.

To continue to promote a phylogenetic perspective in evolutionary biology – The last few decades have seen the realization of the importance of viewing organismal diversity and evolution in the light of phylogeny. This revolution is analogous to and as fundamental to its field as the revolution in cosmology from a Newtonian view of space to an Einsteinian view of space (Maddison and Pérez, 2000). Just as mass curves space, phylogeny has curved the space of biological diversity, providing a distortion on the distribution of traits of organisms we see around us. MacClade and Mesquite are both designed to provide a corrective lens, to help us to see organisms and their traits in their natural orientation within this curved space along the phylogeny. Mesquite's modularity allows this perspective to be extended to fields such as morphometrics, in which a phylogenetic perspective has relatively recently begun to suffuse the field.

Which to use, Mesquite or MacClade?

[Version 4 of MacClade](#) (Maddison & Maddison, 2000) was released in October 2000, and the MacOS X compatible version 4.04 in July 2002. The reader might wonder why we have been working on two different programming efforts, and whether they are intended for different uses. Although Mesquite's extensibility means that eventually it could take on all of the functions of MacClade, in fact for the near future Mesquite will not. Some calculations and functions of MacClade's tree window might not be

available in Mesquite for a while, including particular charts (e.g., Changes and Stasis), equivocal cycling, some of the parsimony options (irreversible, stratigraphic, Dollo), and some options for tree printing (e.g., saving Tree as graphics file or to clipboard). The most significant advances of MacClade 4 over MacClade 3 are in the data editor, where editing of molecular sequences is much more sophisticated, with tools for manual sequence alignment and on-the-fly translation to amino acids. MacClade's data editor might maintain important advantages over that in Mesquite for a while.

In addition, for many of its functions MacClade will remain faster and easier to use than Mesquite. The speed advantage is due primarily to its being in native code instead of Java. MacClade, being a non-extensible program written for a single operating system, has its components more tightly integrated than Mesquite's modules can be, and its user interface tailored for the MacOS. The means that users may find MacClade easier and simpler to use than Mesquite. While we have worked hard to make Mesquite easy to understand and use, its modular nature means it is unlikely to be as simple to the user as MacClade.

Thus, MacClade will continue to be used and useful, even though Mesquite is based on a newer architecture. MacClade has its strengths, and Mesquite will have different strengths. We are using MacClade 4 with our own data (when we get time to work on our own data...), and expect to continue using it indefinitely.

We imagine that in the long-term future MacClade will give way to Mesquite as Mesquite matures. For the next several years, however, the two will coexist and be complementary.






References

Maddison, D.R. and W.P. Maddison. 2000. MacClade version 4: Analysis of phylogeny and character evolution. Sinauer Associates, Sunderland Massachusetts.

Maddison, W. and T. Pérez, 2000. Biodiversidad y lecciones de la historia. In: Enfoques contemporáneos para el estudio de la biodiversidad [Hernández, H.M., A. García Aldrete, F. Álvarez and M. Ulloa, editors]. Instituto de Biología, UNAM, Mexico. Pp. 201-220.

Publishing results from Mesquite

Are Mesquite's calculations well-enough tested to be reliable for published analyses? Because Mesquite is modular, the answer to the question of publication-readiness may not be a simple "yes" or "no". Mesquite modules are marked as either being prerelease versions (not ready for published results), or release versions. In addition, the modules are marked as substantive (possibly involved in producing results), or not (simple graphical or administrative modules not likely to affect results). When all of the substantive modules involved in a calculation are release versions, then we consider it as safe to publish the results as it is with any such biological software. Note however our [disclaimer](#).

How do you know if any substantive modules involved in a calculation are prerelease versions? Mesquite windows currently show either  (green check) or  (red !) in the information bar. The  indicates that at least one module involved in producing the results of the window is marked as both substantive and pre-release; a  indicates that all substantive modules are release versions. Also, if you select the Modules tab of the window to see the modules involved in the window, those modules that are both substantive and pre-release are marked by the .

With the release of Mesquite 1.0, we are indicating that we would be comfortable publishing Mesquite analysis of our own data. We note however our [disclaimer](#), and urge you to pay attention to any strange, unexpected or apparently incorrect behavior of Mesquite, and to send bug reports to us at info@mesquiteproject.org.

How to cite Mesquite?

Citing the system in general

The citation for Mesquite is:

```
Maddison, W. P. and D.R. Maddison. 2003. Mesquite: a modular system for
evolutionary analysis. Version 1.0 http://mesquiteproject.org
```

(The version number listed above might not be up to date. Check the Mesquite Startup window or the Project and Files window when Mesquite is running to find the version you have.)

Citing Mesquite for analyses done

Mesquite's unusual modular nature may give great flexibility in calculations, but it can make it difficult to compose a citation for the calculation of published analyses. Here is a hypothetical example. If the analysis were mostly done by a module written by J. Doe and another by T. Za, one possible citation would be as follows: "The Snidely Index was calculated using the module SNIDIND (Doe, 2003) within the Mesquite system for phylogenetic computing (Maddison and Maddison, 2003); its null distribution was determined by calculating it over 1000 trees simulated by the module Uniform of the SimSpeciation package (Za, 2003) with parameters $s = 0.3$ and $e = 0.1$." with the literature cited indicating:

```
Doe, J. 2003. SNIDIND: a Mesquite module for calculating the Snidely Index,
version 1.2.
```

```
Maddison, W. P. and D.R. Maddison. 2003. Mesquite: a modular system for
evolutionary analysis. Version 1.0 http://mesquiteproject.org.
```

```
Za, T. 2003. SimSpeciation: a package of modules to simulate evolutionary
trees. Version 1.0.
```

How to figure out what modules to cite

A single Mesquite analysis may be the result of the cooperation of many modules, some of which are worth citing (like a module that calculates a key value), some of which are not (like a module that draws the shape of the tree). While we could expect the user to keep track of the calculations requested and what modules to cite, Mesquite has some built-in features to help, via tabs in the information bar of each window. The two tabs that most directly help with citations are:

- **Citations tab:** when touched it shows the citations for modules involved in the analysis. This is the most direct way to find citable modules for an analysis.
- **Parameters tab:** this shows the parameters of the modules. These may include settings such as rates, weights, population sizes, the tree being used, and so on. They can be very important to help you keep track of the assumptions and input behind your results.

Another relevant tab is:

- **Modules tab:** when touched it shows in the window the employee tree of modules involved in producing the window. This includes modules involved in calculations shown. It is useful to help you understand what modules are in use, but it includes all of the modules involved, not just the ones

worth citing.

Which version is being used?

The current version of the Mesquite system being used is shown the in Mesquite window (the window that appears on startup) and in the Projects window (which appears to the left of the screen following startup). The current versions of the modules are reported in the citations view of each the window.

Disclaimer

THIS SOFTWARE IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND. WAYNE MADDISON AND DAVID MADDISON DO NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS REGARDING THE USE OR THE RESULTS OF THE SOFTWARE OR DOCUMENTATION IN TERMS OF THEIR CORRECTNESS, RELIABILITY, CURRENTNESS, OR OTHERWISE. IN NO CASE WILL THESE PARTIES BE LIABLE FOR ANY SPECIAL, INCIDENTAL, CONSEQUENTIAL, OR OTHER DAMAGES THAT MAY RESULT FROM USE OF THIS SOFTWARE

Support & Advice

Because Mesquite has just been released as version 1.0, we expect there will be bugs that appear as new users begin to use it. Please report bugs to info@mesquiteproject.org or to the Mesquite listserv (see below). Some bugs you may encounter will be bugs not in Mesquite but in the operating system or Java virtual machine (for example, see this [list](#)).

Why doesn't Mesquite do <insert your favorite calculation here>?

Mesquite is extensible. What it can do now is a small part of what we envisage. We're aware of many things that could be added. The biggest needs we feel at the moment are for likelihood calculations and more tools for examining character correlations.

We would welcome your suggestions at info@mesquiteproject.org.

Of course, if you really want a new feature sooner than we can add it, you are welcome to write your own module to implement the feature. See the Mesquite website for information on [development](#).

Why is this bug here?

If you think you have found a bug in Mesquite, please [report](#) it to us (see [below](#)). If we don't know about a bug, it is unlikely that it will be fixed. It might be natural to presume that we know about all the bugs in Mesquite, but it is not possible for us to have tried out every combination of options on every computer environment with every data matrix, model, etc. If you are not sure if it is a bug (a large number of users blame themselves when they encounter a bug, presuming it is something they have done wrong), please report it anyway. Even if your apparent "bug" is a mistake you made, your mistake might indicate a poorly designed interface that should be corrected.

That there will be bugs in Mesquite is inevitable. Mesquite is a large project, about 1500 pages of source code when printed single spaced at 7 point font (about 120,000 or more lines of source; over 5MB of ASCII text). We don't have a big team of people working on it; it was initially written by a single person, and now two. We apologize for the bugs that remain, and look forward to your bug reports and efforts to aid us make Mesquite useful.

Mesquite is intended to become a community effort. Source code is posted on the [Mesquite web site](#); we hope other programmers will help us improve the system.



Check out the Help system first

Mesquite doesn't have conventional documentation for all of its features, but it does have various features to help you learn how to use it. Check out the [help page](#) for instructions. Also, browse through the example files, as they contain many explanations and hints.

We recognize that Mesquite is a complex system that offers many choices to the user. We plan to build macros and other tools to help build paradigmatic analyses for users who don't want to sort their way through options every time. However, they're only beginning to be built.

Reporting bugs and requesting advice

Partly to foster a community of users, and partly for efficiency, we are encouraging users to sign up to a LISTSERV via which they can report bugs and post queries about using Mesquite. You may report bugs and ask for Mesquite advice via this LISTSERV. To sign up for the listserv, send an email with the following message in the body of the email:

subscribe MESQUITELIST YourFirstName YourSecondName

to

listserv@listserv.arizona.edu

where you replace "YourFirstName" by your first name, and "YourSecondName" by your second name. Once you've signed up, you can send a message to the list.

If you're reporting a bug, it's important that you are able to specify, as precisely as possible, exactly what you did that generated the bug. Try to find a precisely repeatable series of actions that generate the bug.

In any bug report, please specify the version of Mesquite and any important modules and your operating system. Be prepared to send us your data file so we can test the problem.

(Of course, we recognize that some messages may be more appropriately directed directly to us. You may contact us at info@mesquiteproject.org).

Java bugs that affect Mesquite

Below are problems in Java on various operating systems that affect Mesquite. If these problems annoy you, please contact those responsible for your Java virtual machine.

Mac OS 9

- The cursor will not change to reflect the current tool. This is a limitation of the version of Java used on MacOS 9.
- There may be a rare, innocent exception thrown.

MacOS X 10.0 through 10.1.5

- If a pop-up menu appears, the colored spinning cursor may appear and prevent you from choosing a menu item for several seconds. You may lose control for 5-10 seconds. If you move the mouse slightly, you will regain control more quickly.
- Text in dialog boxes is sometimes only partially visible. This is a bug in MacOS Java 1.3.1 Update 1.
- Windows near top of screen may be unresponsive (workaround: move window down or resize). This is a bug in MacOS Java 1.3.1 Update 1.
- Various exceptions are thrown. (You will get a little window noting various problems when this happens.) These are bugs in MacOS Java 1.3.1 Update 1.
- Double clicking a file's icon in the Finder when Mesquite is already running will open the file. However, if Mesquite is not yet running, Mesquite will start but will fail to open the file.

MacOS X 10.2 ("Jaguar") running Java 1.3.1

In general Mesquite works well under Jaguar with Java 1.3.1, except for being slow in screen display, the annoying dialog box problems, and, on some machines, a problem with chart display.

- Dialog boxes occasionally fail to show items (sometimes they are totally blank!). Dialog boxes might also appear in the incorrect location, or with items drawn double. If you wait for a moment, the dialog should reappear correctly.
- Various exceptions are thrown. (You may get a little window noting various problems when this happens.) These seem mostly innocent.
- Occasionally when you try to drag a branch in the tree window extra lines are left littering the screen.
- On some machines, a virtual machine bug causes the graphics view of the chart window to stay visible even after switching to the text or other views. This blocks one's ability to see the text or other information. We have a workaround for this problem; as this workaround itself exposes other virtual machine bugs, we have not released it. However, if you work with charts a lot, you may want a copy of this workaround; please contact us if you do.

MacOS X 10.2 ("Jaguar") running Java 1.4.1

We do not yet support Mesquite running under Mac OS X using the Java 1.4.1 virtual machine, which has many bugs. The latest software from Apple includes both Java 1.3.1 and Java 1.4.1; Mesquite automatically requests use of 1.3.1.

Windows, Java up to and including 1.4.0

- Windows bounce up and down as menus change (a rapid shrink & expansion). This is due to a design flaw either in Windows or the Java VM; there is nothing we can do about it.
- After Mesquite is used for a while, about 5% of menu labels are randomly scrambled with labels of other menu items. For instance, the File menu might be labeled as the "Gray" menu, or the "Histogram" menu, or have some other label drawn "randomly" from among the menu items within the menus. This is a bug in Sun's virtual machine (Sun has refused to consider fixing this because we have been unable to reproduce it in a smaller program.) Because of this problem and the confusion it might cause, we have added a Reset Menus item in the File menu. This will force Mesquite to rebuild the menus, which usually corrects the mislabeling.

Linux/Unix etc

- Window sizes and placements sometimes are inappropriate under some window managers. Some window managers refuse to let Mesquite have control over window size and placement, and choose what appear to be random sizes and placements of windows. This is not in our control. Try changing your window manager. We have found that the default installations for KDE and Gnome of Red Hat 7

work well with Mesquite.

- Messages about Fonts not found on startup will be given commonly, in fact we suspect on most installations on Linux and Solaris of Sun's Java VM prior to 1.4. The problem is that you need to install a supposedly optional package of fonts in order that the default installation of Sun's Java VM work properly (see <http://java.sun.com/products/jdk/1.2/changes.html#sunw>). We would have thought that if the default installation of a product claims to support a program but doesn't, then that would be considered a bug. Sun considered this to be not a bug, although they seem to have fixed it...
 - Dialog boxes sometimes appear very small, too small to be used. This may be due to a bug in the virtual machine of Java 1.3 or earlier. Try running Mesquite with Java 1.4 or higher. If this does not fix the dialog box sizing problem, please report the bug to us.
-

Credits

Mesquite was begun in 1997, but its roots go deeper, to the initial development in 1985 of [MacClade](#), which even from the start allowed interactive manipulation of trees and interpretation of character evolution. As features were added to MacClade through the years, most from Wayne Maddison in early versions (1 and 2) and from David Maddison in later versions (2, 3 and 4), MacClade developed an exploratory approach to phylogenetic calculations with a distinctive user interface. After MacClade version 3 was released in 1992, the coauthors worked to ready version 4. Almost all of the many new features in version 4, including the new facilities for molecular sequence editing, were the results of David's efforts. Wayne's efforts on MacClade 4 involved an attempt to graft a modular architecture onto MacClade to allow plug-ins so that its capabilities could be extended. This, we hoped, would allow us and other programmers to add many new tree-based analyses to MacClade. After about a year of work on this, it became clear that grafting this new architecture on to an existing program was not going to work. MacClade was then returned to its original, non-modular state, and it was within this more traditional framework that David completed MacClade 4.

In order to build the desired modular architecture, Wayne had to start from scratch, and so a new project was born in July of 1997. The very first prototype, after one day of work, can be seen [here](#) (for the first few days it was called BeanTree, before it became known as Mesquite). Mesquite's vision, exploratory nature, and its user interface borrow extensively from ideas developed in MacClade, but the underlying architecture is quite different. Thus, Mesquite contains a mix of features borrowed directly from MacClade, features we had wanted to put into MacClade but couldn't (e.g., coordinated selection of objects, Trace Character over Trees, likelihood reconstructions), and newly conceived features.



The subsequent chronology of the Mesquite project is:

- August 1998 - first public demonstration (Cambridge University)
- July 1999 - limited seeding to a few developers (prototype version 0.9.2)
- August 1999 - project web page on-line (currently at <http://mesquiteproject.org/>)
- 29 September 1999 - broader release to developers (prototype version 0.9.5)
- early 2000 - passed 100,000 lines of total code, and 200 total modules.
- 26 June 2000 - Mesquite introduced at Evolution meetings, Bloomington, Indiana (prototype version 0.9.28)
- 14 March 2001 - first public beta version (version 0.95.80)
- 2 April 2001 - public beta version (version 0.96)
- 24 July 2001 - version 0.98 with source code released
- 21 August 2002 - version 0.99 released
- 14 September 2002 - version 0.991 released
- 27 September 2002 - version 0.992 released (internal version (build) d24)
- 10 January 2003 - version 0.993 released (build d42)
- 7 February 2003 - version 0.994 released (build d51)
- 21 May 2003 - version 0.995 released (build e23)
- 21 June 2003 - version 0.996 released (build e30)
- 22 September 2003 - **version 1.0 released** (build e58)

From July 1997 through October 2000, the architectural design, programming and documentation for the basic Mesquite libraries and modules was done by Wayne Maddison, with occasional input from David Maddison. David entered the project in earnest in November 2000. (Other packages of modules for the Mesquite system are due to other authors: for instance, the Rhetenor package of morphometrics modules is by Eric Dyreson and Wayne Maddison.)

Acknowledgments

Mesquite was developed with the assistance of a Fellowship to WPM from the David and Lucile Packard Foundation, and the patience of our families.

David Swofford helped with the implementation of the likelihood calculations, and supplied code for the optimization routines. Lars Rosengreen helped us prepare the code for compilation by means other than Codewarrior.

For feedback, including bug reports, and other assistance we thank Peter Midford, Michel Laurin, Lars Rosengreen, Cymon Cox, Mario Cozzuol, Korbinian Strimmer, Alexei Drummond, Lacey Knowles, Jonathan

Coddington, Alan de Queiroz, Robin Allanby, Stephanie Diezmann, François Lutzoni, Jolanta Miadlikowska, Frank Kauff, Matt Hare, Jennifer Steinbachs, Margaret Thayer, Lukas Ruber, Olga Zhaxybayeva, and Galina Glazko.

Mesquite Help: Learning how to use Mesquite

Why Mesquite is complex

Mesquite has many options depending on what modules are installed and loaded. Its modularity and flexibility allow for many possible analyses, but also create challenges for the user (and the manual writer). There are too many possible analyses for us to have yet written instructions specifically for each. If the user wants to perform some particular analysis, he or she may have to use his or her puzzle-solving ability to figure out how to achieve the analysis by combining Mesquite's various functions.

Why are so many choices? Why do some analyses assault you with a guantlet of many dialogs? These are consequences of Mesquite's flexibility. We have tried to protect you from complexity as much as possible.

You may find a slightly different rhythm of thinking will be needed when confronting Mesquite. For instance, suppose you wanted to see how much the likelihood for a character varies if random noise is added to the branch lengths of the phylogeny. If you were lucky, there would be a macro or some menu item that would build your desired analysis directly, but for this question there isn't currently such a shortcut. What you'd like to see is a frequency distribution of likelihoods calculated for the character over a series of trees, each tree derived from some given tree by adding random noise to the branch lengths. This sounds like a histogram, but which histogram? It seems to concern characters, so perhaps it is a Characters histogram? No, each sample point is a tree, and thus you should ask for a Trees histogram. When a dialog box asks you what value to calculate for the trees, what do you say? The likelihood concerns a character, and thus you respond "Tree value using character". In the next dialog, you can choose character likelihood. You will also be asked what source of trees. The trees are randomly modified by adding noise to the branch lengths; therefore, choose "Randomly Modify Tree" and for the particular modification, "Add noise to branch lengths". Although this may seem complex, it does allow you to design the analysis exactly as you wish. You may use the example files, instructions in this and other manuals, and the [Mesquite discussion list](#) in order to learn how to do the analyses you need.

Features that help reduce Mesquite's complexity to the user are:

- **Primary Choices:** Submenus and dialogs boxes with lists of alternative choices may have a small number of choices listed then an item "Other Choices..." (if a submenu) or a check box "Show Secondary Choices" (if a dialog box). If these are selected, a larger number of choices is offered, including the "secondary" choices. This division of choices into the primary choices (the ones we expect will be most frequently selected) versus the secondary choices helps keep the commonly seen array of choices small. (This division can be turned off in the Defaults menu of the Project and Files window or the Log window.)
- **Macros:** Mesquite can be instructed by a [scripting language](#). Macros can therefore be written or automatically generated to script complex calculations. Macros appear in submenus in the appropriate menu.
- **Configurations:** Even if many packages of Mesquite modules are installed, you can ask Mesquite to load only a subset of them. This allows Mesquite to startup more quickly and to present a simpler interface (i.e., with fewer options). You can control configurations using the submenu [File>Activate/Deactivate Packages>](#).

Mesquite's documentation and other learning aids

While we hope that users will always apply their imaginations in using Mesquite, we have attempted to make Mesquite as easy as possible to understand via the following:

Manual: This manual provides an introduction to Mesquite, including instructions for accomplishing some analyses. It resides in the **docs** subdirectory of the **Mesquite_Folder** directory, and is also available from the Help menu while Mesquite is running.

Additional web pages: Mesquite surveys modules for information and composes a set of web pages. These web pages list all the modules loaded, brief explanations as to what these modules do, and the scripting commands these modules respond to. These Mesquite-composed HTML pages are in a directory called "Mesquite_Prefs" within a directory **Mesquite_Support_Files**, which may reside in different places depending on your operating system. If you can find these pages, you might want to store a bookmark, or alias, or shortcut to one of them so that you can find them again without going through Mesquite.

Example files: One of the best ways to learn about analyses is via the example files, which are distributed with the main Mesquite package and with various of the add-on packages. The example files are present in the "examples" folder of Mesquite_Folder. Some additional examples are outlined on the [Studies](#) page.

Explanation areas: [Explanation areas](#) at the bottom of each window may describe the window, its contents, or the function of a selected button or other object. If you hold down the Shift key as you select a menu item, an explanation for it will (usually) appear in the explanation area of the frontmost window. Explanations areas are also present in some dialog boxes.

Menu & Control Explanations: The menu item [Window>Menu & Control Explanations](#) causes Mesquite to compose a HTML page summarizing the menu items or buttons for the current window, and to show it to you.

Keyword Search: This menu item in the Help menu of Mesquite provides a currently-primitive facility to search among the names and explanations of all of the installed and loaded modules to find a keyword. You could, for instance, search for "simulat" to find all of the modules that might have to do with simulations.

Window information bar: The [tabs](#) at the top of each window allow you to select a view: Graphics, Text, Parameters, Modules and Citations. The Modules view in particular can help you learn about Mesquite calculations. It shows the set of modules currently active in the analysis and or graphics shown in the window. This is shown as a tree of modules employing other modules (a bureaucratic hierarchy!). If you pass the cursor over a module name, an explanation for it appears in the explanation area at the bottom of the window. If you touch on the name of a module, a menu will appear with choices to show more information (if available). If there is a special manual for the module, an extra label will appear that will link you to the manual.

Thus, if you want to learn about:

- **Modules:**
 - Use the menu items in the **Help** menu to go to the appropriate HTML pages for modules or packages. You can also get to the HTML pages of packages by touching the banners in the Mesquite startup window (the window named "About Mesquite").
 - Use the **Keyword search** facility in the Help menu
 - Open the **Modules** view of the window of concern (using the tab in the information toolbar) and move the cursor over the names of the modules to see explanations in the explanation area, or touch on their names to go to their information pages composed by Mesquite.
- **Menu items**
 - Hold down Shift as you select a menu item to make an explanation for it appear in the explanation area of the frontmost window.
 - Use the **Menu & Control Explanations** menu item of the **Window** menu to make a web page summarizing the functioning of the menus and buttons for the foremost window.
- **Buttons and tools:**
 - Touch on a tool in the tool palette to make an explanation for it appear in the explanation area of the frontmost window.
 - Use the **Menu & Control Explanations** menu item of the **Window** menu to make a web page summarizing the functioning of the menus and buttons for the foremost window

How to remember or document what you have already done?

With an interactive program having as many options as Mesquite, it can be difficult to remember what options are currently in effect. Three facilities help you keep track of what you've done.

- Information bar of windows. The [information bar](#) has various tabs that control alternative views of a window's contents. Some of these give information about the current calculations, the parameters in use, and details about the modules in use by the window.
 - The Log Window, available by selecting **Mesquite Log** in the **Window** menu, records commands given and messages relayed to the user. This text is also saved automatically to a file called "Mesquite Log" within the directory Mesquite_Support_Files.
 - Auto-scripting for file saving. When Mesquite saves NEXUS files, it automatically constructs a script that attempts to return an analysis to its current state. This not only allows a user to save a snapshot of an analysis, but the script itself can also be inspected to determine current parameters (in case that's not evident otherwise). Snapshot scripts can also be seen for individual windows, by selecting the appropriate item in the **Scripting** submenu of the **Window** menu.
-

Making, opening and saving data files

Mesquite is currently designed for data files following the NEXUS format (Maddison, D.R., D.L. Swofford, and W.P. Maddison. 1997. NEXUS: An extensible file format for systematic information. *Systematic Biology* 46: 590-621) although it can import and export files of other formats. Thus, you could create your data file with a text editor or word processor if you followed NEXUS conventions. However, you'll probably find it easier to use Mesquite's data matrix editors, tree windows, and so on, to specify the information in the data file, and let Mesquite handle the formatting issues.

Mesquite can read NEXUS files created with [MacClade](#), and can save files that MacClade understands.

Creating a new data file

To create a new blank data file, choose **File>New** (i.e., the New menu item in the File menu). You'll be presented with a dialog box in which you enter the name of the set of taxa (e.g., "Drosophila") the initial number of taxa, whether or not you want to show a tree window, and whether to make a character data matrix. (The taxa could be species, or sequences, or whatever are your "terminal taxa", "Operational Taxonomic Units", or evolutionary units.) (You can just leave the name of the set of taxa as "Untitled" if you wish, but that may become confusing if you ever have more than one set of taxa in the same file.) You can later add more taxa using **(List of Taxa)List>Add Taxa** or **(Character Matrix Editor)>Matrix>Add Taxa**, or by using the Add Taxa tool in the [Character Matrix editor](#).

When you make a new data file, you'll be presented with a list of taxa or perhaps a tree window. The taxa are automatically named "taxon 1", "taxon 2", and so on. You can rename a taxon name (e.g., "D. melanogaster", "D. willistoni", and so on) by selecting the I-beam tool in the List of Taxa window and touching it on the taxon name. There is a submenu, **(List of Taxa)List>Alter Taxon Names>**, that might offer other ways to edit taxon names. Taxon names can also be edited in the [Character Matrix editor](#).

A new data file does not automatically include a matrix of character data unless you request it. To add new matrices, see the section on the [Character Matrix editor](#).

Opening an existing data file

To open an existing data file, use **File>Open>File**. If Mesquite detects that the file is not a NEXUS file, it will ask to you to choose its file format for importing.

Saving a data file

Save a data file using **File>Save File** or **File>Save File As**. You can also export to other formats using **File>Export**.

Projects and files

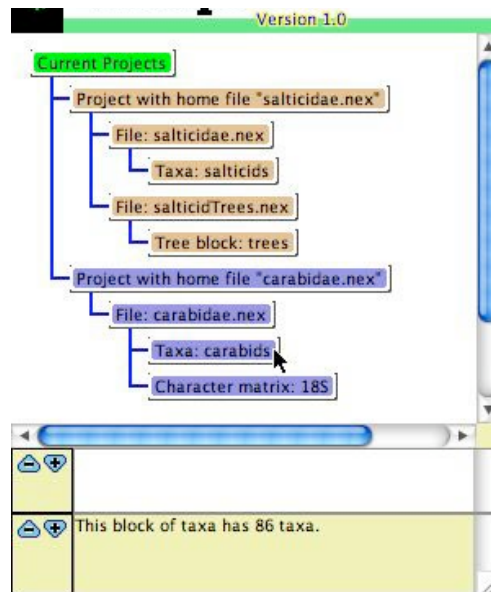
Mesquite is not restricted to considering only one file at a time, but instead can collect information from various files and consider it together. Such a collection of files sharing information is called a **project**. Implicitly when you first open a file, a project is created, one which contains only a single file. Other files can be linked into the project using the Link commands.

Since Mesquite can accumulate and analyze a more or less indefinitely large collection of elements of information (several sets of taxa, data matrices, and so on), Mesquite doesn't need to respect the boundaries of files. That is, it could read a TAXA block from one file on the disk, and read a data matrix for those taxa from another file on disk, and a set of trees from another file. While other programs can handle external treefiles or command files, Mesquite can handle external character matrices, assumptions, and so on.

Mesquite therefore makes a distinction between the collection of elements of information that are currently interacting with one another in Mesquite's calculations, and the physical files on disk or server. The former collection, which may include information gathered from several files, is called a project. The set of files to which the elements of information in a project belong are said to be **linked**.

Mesquite shows a list of the projects and files currently active in the Projects and Files window. In the example shown here, there are two projects currently open. The first, marked by a sandy color, includes two linked files, "salticidae.nex" with a block of taxa and "salticidTrees.nex" with trees. The second project, marked in blue, includes one file "carabidae.nex" with a block of taxa and a character matrix. The windows (tree windows, data editors, etc.) belonging to these two projects will also be distinguished by these different background colors. **Information is not shared between projects**, so that when you are using the windows of "salticidae.nex", the character matrix "18S" will not be available for use.





Opening versus Including versus Linking files

In Mesquite there are three ways to read a file: Open, Include and Link.

1. **Open** — If a file is to be opened up independently of any other open files, and not share information, it should be opened using the Open File... menu item. A file opened in this way is treated as belonging to a project separate from any other files open at the time. In the graphic above, Open File... was called twice, once to open salticidae.nex, second to open carabidae.nex.
2. **Include** — To read in the contents of a file and merge them into an existing file, so that all of its information becomes part of the existing file and is saved into that existing file, use the Include File... menu item.
3. **Link** — To read in the contents of a file and add its information to a collection of information in a project in use, but to maintain the file separate for purposes of writing to the disk, use the Link File... menu item in the File menu. A file opened in this way will become part of the project. Because of the interdependencies among elements of information that can exist (list of taxa in a data file matches list of taxa in a tree file), it is possible that editing information that belongs to one file will also change information in a linked file. In the graphic above, Link File... was called once, to link salticidaeTrees.nex with salticidae.nex.

Mesquite's menus

Mesquite's menus differ from window to window and vary depending on the calculations being performed. In operating systems such as Windows and Linux, each window has its own set of menus embedded at the top, thus making it easy for the user to understand the context — a particular window — in which the menu item exists. Users of the Macintosh OS will see the screen menu bar change depending on what window is at the forefront. On the other hand, there is considerable duplication of menu items from window to window, which reflects the fact that windows share context to varying degrees, but which may be confusing to users of operating systems with window-specific menus. We will not here take sides about which menu paradigm is better, but note that Mesquite's menu are firmly organized so as to reflect the heirarchical structure of the calculations performed.

1. [Learning about menus](#)
2. [Standard menu items](#)
3. [Where to find menu items](#)

Learning about menus

Mesquite menus include a few constant ones (File, Edit, Window) and various other menus that change as different modules are used. This situation could be confusing to the user. For instance, menu items for particular modules may appear in different places depending on the context.

Below is a list of the standard menu items that should appear in most configurations of Mesquite. There will likely be other menus and menu items in addition to these.

Mesquite can automatically generate documentation for the current menu configuration and present it to the user as a web page. This can be done by selecting the **Menu & Control Explanations** menu item from the **Window** menu of Mesquite. Also, if you hold down the Shift key while selecting a menu item, an explanation of the menu item will appear in the explanation are of the foremost window.

Standard menu items

File

- **New** — Make a new file (as a new project)
- **New Linked File** — Make a new file (linked to the current project)
- **Open File...** — Open file as a separate project (not linked to current project)
- **Open External**
 - **URL...** — Get file from web server
- **Link File...** — Open file and incorporate into the current file (or home file of current project)
- **Include File...** — Open file as linked (and hence sharing information with) the indicated project
- **Close File** — Close the file
- **Close All** — Close all projects
- **Close Window** — Close the foremost window
- **Save File** — Save the file to disk.
- **Save File As...** — Save the file to disk under another name.
- **Export...** — Save the file to disk under another name and with a format other than NEXUS.
- **Save Window As Text...** — Save the text version of the window (visible by touching on the Text tab of the window).
- **Print Window...** — Print the foremost window.
- **Activate/Deactivate Packages** — Change the configuration of modules loaded. These menu items affect what modules, among those installed in your copy of Mesquite, are actually loaded at startup. Most configurations load only a subset of menu items, for instance to simplify the interface.
 - **Use All Installed modules** — On next startup, load all installed modules
 - **Choose Configuration** — Choose the configuration of module packages loaded at next startup.
 - **Delete Configuration** — Delete user-defined configurations
 - **Define Configuration** — Define a new configuration of modules to be loaded at startup
- **Macros**
 - **Show Macro List** — Show list of Macros available, and where available explanations as to what each does.
 - **Edit Macro Information** — Change name and explanation for user-defined macros.
 - **Show Macro List** — Show list of Macros available, and where available explanations as to what each does.
- **Rename Log file** — Renames the current log file.
- **Reset Menus** — Forces rebuild of all menus. Useful on Windows because of a bug in the Java virtual machine that occasionally scrambles menu names
- **Force Quit** — Forces Mesquite to quit without the usual clean-up and checking whether the file has unsaved changes. Should be used only in emergency situations (e.g. if some calculation appears to have crashed)
- **Quit Mesquite** — Quit Mesquite

Edit

- **Undo** – Minimally functional. Undos most recent change to tree in tree window. Not available for Character Matrix Editor and other windows
- **Cut, Copy, Paste** – Have their standard uses, though currently only deal with text.
- **Font** – Change the default font for the foremost window.
- **Font Size** – Change the default font size for the foremost window.
- **New Generic Nexus Block** – Make and edit a NEXUS block for the file.
- **Edit Mesquite script** – Edit a Mesquite script contained in the file
- **Edit Comment** – Edit the comment about the file.

Characters

- **Character Matrix Editor** ▶ – Show a data editor for a character matrix
- **List of Characters** ▶ – Show the list of characters for a particular character data matrix
- **New empty matrix...** – Make a new, blank character data matrix
- **Make New Matrix from** ▶ – Make a new character data matrix by copying an existing matrix or filling the matrix from a special source (e.g., simulated characters)
 - sources of matrices listed here
- **List of Character Matrices** – Show the list of available character data matrices
- **List of Character Models** – Show the list of available models of character evolution
- **New Character Model...** ▶ – Make a new model of character evolution
 - types of character models listed here
- **Edit Character Model...** ▶ – Edit an existing model of character evolution
- **Lists**
 - **Parsimony model sets** ▶ – Show a list of the sets of assignments of parsimony models for a particular character data matrix
 - **Probability model sets** ▶ – Show a list of the sets of assignments of probability models for a particular character data matrix
 - **Character sets** ▶ – Show the character sets for a particular character data matrix
 - **Inclusion sets** ▶ – Show the character inclusions sets for a particular character data matrix
 - **Character Partitions** ▶ – Show the character partitions for a particular character data matrix

Taxa&Trees

- **New Tree Window** ▶ – Show a new tree window for a particular set of taxa (multiple tree windows can be shown)
- **Current Tree Window** ▶ – Bring an existing tree window to the front
- **Multi Tree Window** – View several trees at once in window.
- **List of Trees** ▶ – Show a list of trees in a tree block in the file
- **New Empty Block of Trees...** – Make a new empty block in which to store trees
- **Make New Trees Block from** – Make a new block of trees by copying an existing block or filling the block from a special source (e.g., simulated trees)
 - available sources of trees listed here
- **List of Tree Blocks** – Show a list of tree blocks in the project
- **List of Taxa** ▶ – Show a list of taxa for a particular set of taxa
- **New Block of Taxa...** – Make a new set of taxa
- **List of Taxa Blocks** – Show a list of the different sets of taxa
- **List of Taxa Partitions** – Show a list of taxa partitions that exist
- **List of Taxon Sets** – Show a list of the Taxon sets that exist

Analysis

- Menu items to make new charts and other analyses are listed here

Window

- **Show Information Bar** – Show or hide the information bar at the top of the window
- **Menu & Control Explanations** – Causes Mesquite to compose a web page listing the menu items currently in the menu bar for this window, and where available gives explanations for what they do. Also lists the buttons currently in the window, and where available gives explanations for what they do.
- **Clone Window** – Clones the foremost window. Useful to reproduce an existing analysis then change it slightly. Cloned window will appear exactly above original window; thus the clone should be moved to view the original.
- **Save Window As Macro** – Save foremost window as a macro. This macro can later be used to reproduce the window's appearance and analyses.
- **Macros** – Execute the selected macro.
- **Scripting** – The "Show Snapshot" item displays the script that would be required to return the

- **foremost window to its current states;** "Send Script" sends a script to the foremost window.
- **Bring All Windows To Front** – Brings all Mesquite Windows to the front.
- **Current Windows** – Allows you to select current Mesquite windows to bring them to the front.
- **Mesquite Startup Window** – Makes the Mesquite startup window visible.
- **Mesquite log** – Makes the Mesquite log window visible.
- **Mesquite Projects and Files** – Makes the Mesquite projects and files window visible.
- **License** – Shows the license under which Mesquite is distributed.

Help

- **Mesquite Manual** – Show home page of Mesquite manual
- **Modules Loaded** – Show web page of modules that are installed and loaded
- **Keyword Search** – Search for exact keyword among the names and explanations of loaded modules.
- **Scripting Commands** – Show web page of available commands for scripting
- **Active Module Tree** – Show the entire employee tree of modules
- **List Active Modules** – Write to the log a list of the entire employee tree of modules
- **List Prerelease Modules** – Write to the log a list of any modules loaded that are labelled as prerelease and substantive
- **Show Developer Statistics** – Write to the log some information useful for debugging.

Where to find menu items

Other menus come and go depending on which window is in front and what calculations are in use. Thus, the standard Tree Window has a Tree menu.

As the user requests calculations and output of results, the menus change. The reason for this is that different menus are appropriate in different circumstances. Thus, if the user requests the Balls and Sticks method to draw trees, a menu item "Spot size" appears in the Tree menu. Spot size controls the size of the balls drawn at the nodes of the tree.

Exactly where a menu item appears may at first be confusing to the user, but there is a certain logic to it that hopefully won't be too obscure (technical details are available in the [developer's documentation](#)). The "Spot size" menu item belongs to the module that draws trees in the Balls and Sticks fashion, since that module wants the user to be able to control the size of the balls. That menu item appears within the menu of the modules that employ it to draw trees. Since the Drawing menu belongs to the module that coordinates tree drawing, and it is this module that (indirectly) is employing Balls and Sticks to draw the tree, the menu items belonging to Balls and Sticks appear in the Drawing window.

A module can also have its own menu, as does the tree drawing coordinator. Thus, the menu items of a module (usually) appear in the menu belonging to its closest employer module that has its own menu. We say "closest" because sometimes a module will have a menu item, but its immediate employer won't have a menu to put it in. The menu item drifts upward in the employment hierarchy until it finds an employer that has a menu that can house it.

This system means that menu items appear in an appropriate context, but that that context is not fixed because it depends on what employer has a menu in which the item can be housed.

Mesquite's Windows

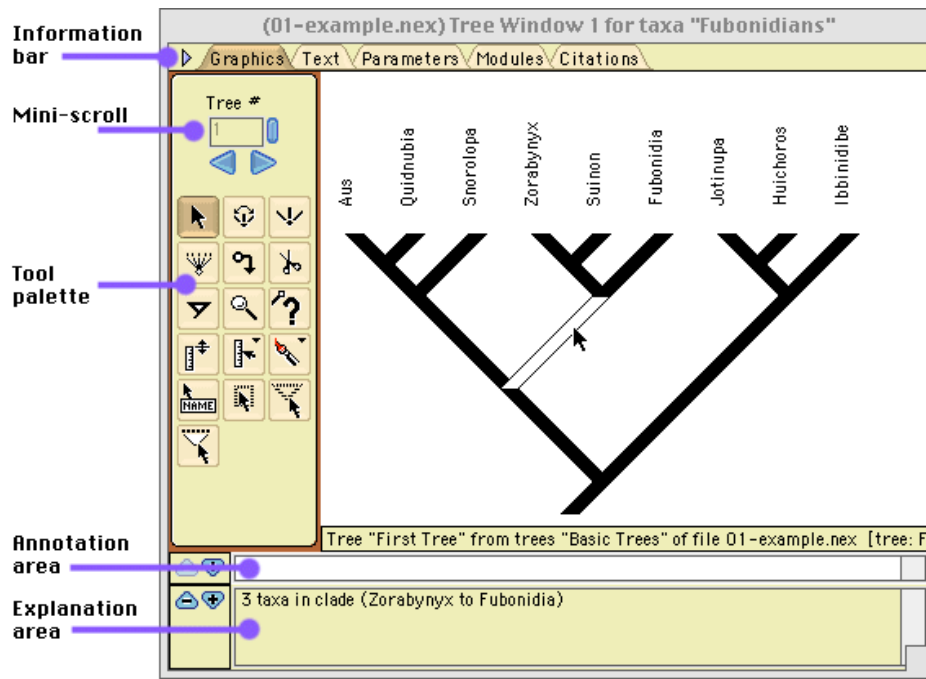
Some basic Mesquite windows are:

- **Startup window** - the colorful window with the mesquite leaf, the progress bar indicating modules loading, and the banners at right for installed and loaded packages.
- **Mesquite log** - the text window showing the log of Mesquite activity
- [Project and files](#) - the window at left showing open projects/files and their components
- [Tree Window](#) - the window to view and edit trees
- [Character Matrix Editor](#) - a spreadsheet editor for character matrices
- [List Windows](#) - show lists of characters, taxa, trees, models and other objects.

On this page are explanations of the general features of windows, as well as issues specific to list windows.

Common features of windows

Windows in Mesquite have a consistent structure, with these components commonly appearing:



Here are brief descriptions of some of these:

- [Information bar](#)
- [Mini-scrolls](#)
- [Tool palettes](#)
- [Annotation area](#)
- [Explanation area](#)

Window information bar

Windows can have an information bar that allows you to access information about the window, its calculations, and the modules involved in them. If the information bar is not showing and you wish to see it, select "Show Information bar" in the **Window** menu. The information bar can be turned off by clicking the triangle at its left.



The tabs in the information bar choose the different modes of display of the window. The first two of these (the graphics tab, and the text version tab), cause the window to display its basic output (e.g., the spreadsheet data editor, the tree drawn in a tree window) in either a graphics form or a textual form. The remaining tabs give information of other sorts.

The window modes are:

- **Graphical version of output** — The standard output of the window showing results and analyses.
- **Textual version of output** — A textual version of the results and analyses.

- **Parameters of modules** — A list of the current settings and parameters of the modules involved in producing the window and its contained results.
- **Tree of Modules** — Shows the employee tree of all modules participating in the window
- **Citations for modules** — Gives citations for some of the modules participating in the window.

Mini scrolls

Mini scrolls are used to scroll among trees, characters and other items. The small blue button right of the text edit box, when hit, tells the scroll to enter and use the number within the edit box. Otherwise you can use the arrows to scroll forward or back. If the change is not acceptable (e.g., because you are already at the minimum or maximum value) the arrow is dimmed.



Tool palettes

In various windows are tool palettes that look something like this:



The button of the current tool is shown darker. Usually in the explanation area of the window, an explanation is shown for a tool when you click on it. Some tools (whose buttons may show a small inverted triangle) when double clicked, right-clicked or control-clicked, show a menu in which options can be chosen.

Annotation area

The white annotation area shows footnotes or other annotations stored for taxa, trees, characters or other objects. These annotations are sometimes editable. At its left are two buttons, a minus and a plus arrow. These reduce and increase the height of the annotation area.

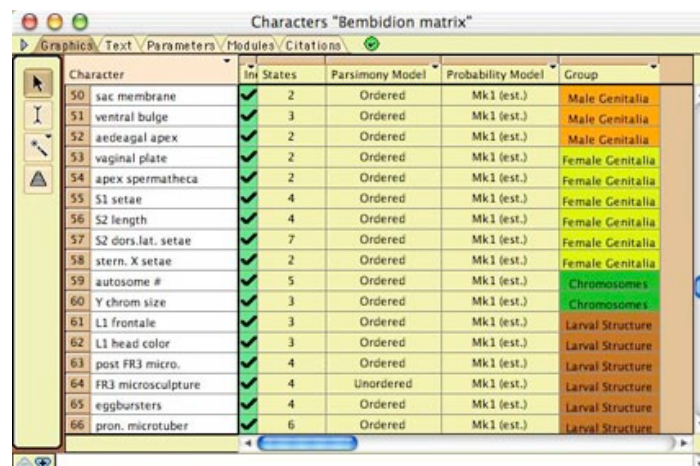


Explanation area

The explanation area, below the annotation area, shows explanations for tools, modules, objects in files, and so on. Its height can be controlled by the minus and plus arrows.

List Windows

In the Characters and the Taxa&Trees menus are available various windows that give lists of objects with information about each, including the List of Characters and the List of Taxa Windows. Other List Windows present lists of tree blocks, character matrices, parsimony models, probability model sets, and so on. These List Windows have a consistent interface: you can select rows and columns, show and hide columns, and possibly rename and delete objects.



Character	Ini States	Parsimony Model	Probability Model	Group
50 sac membrane	2	Ordered	Mk1 (est.)	Male Genitalia
51 ventral bulge	3	Ordered	Mk1 (est.)	Male Genitalia
52 aedeagal apex	2	Ordered	Mk1 (est.)	Male Genitalia
53 vaginal plate	2	Ordered	Mk1 (est.)	Female Genitalia
54 apex spermatheca	2	Ordered	Mk1 (est.)	Female Genitalia
55 S1 setae	4	Ordered	Mk1 (est.)	Female Genitalia
56 S2 length	4	Ordered	Mk1 (est.)	Female Genitalia
57 S2 dors.lat. setae	7	Ordered	Mk1 (est.)	Female Genitalia
58 stern. X setae	2	Ordered	Mk1 (est.)	Female Genitalia
59 autosome #	5	Ordered	Mk1 (est.)	Chromosomes
60 Y chrom size	3	Ordered	Mk1 (est.)	Chromosomes
61 L1 frontale	3	Ordered	Mk1 (est.)	Larval Structure
62 L1 head color	3	Ordered	Mk1 (est.)	Larval Structure
63 post FR3 micro.	4	Ordered	Mk1 (est.)	Larval Structure
64 FR3 microsculpture	4	Unordered	Mk1 (est.)	Larval Structure
65 eggburststers	4	Ordered	Mk1 (est.)	Larval Structure
66 pron. microtuber	6	Ordered	Mk1 (est.)	Larval Structure



At left of the List Window is a tool palette. The **arrow tool** allows you to select columns or rows. The **I-beam tool** allows you to edit the name of an object. The **magic wand tool** selects all rows that have the same or similar value in the column touched. The magic wand tool has alternative selection criteria that can be chosen by clicking and holding the cursor on the tool in the tool palette. The **sort tool** reorders the objects listed according the column on which you touch. Thus, if you touch on the leftmost column showing names, the objects would be sorted alphabetically. If you touch on a column showing a numerical result, the objects are sorted in ascending or descending order of the number. The sort tool sorts in ascending order by default, but by descending order if the option or ALT key is held down.

Each List Window has a List menu, with which you can show or hide columns. Some columns merely inform you of some quality of the object. Others may represent analyses. For instance, the column may show the value of some statistic for each of the characters listed in the List of Characters window.

For some of the List Windows, you can ask to delete the objects whose rows are selected by choosing "Delete Selected" from the List menu.

Mesquite's Charts

Charts compile and display values for a series of objects or items, whether taxa, trees, characters, or matrices. Mesquite has two primary styles of charts:

- **Histogram** - a bar chart summarizing frequencies of values in different categories for each of many objects.
- **Scattergram** - a plot showing each object's value in two variables

Most of Mesquite's charts are available through the first few submenus of the Analysis menu.

We suspect the greatest challenge to the user will be learning what chart to choose and how to set it up. For instance, how can one create a chart summarizing the estimated rate of a character's evolution according to each of a series of trees? Does one choose [New Histogram For>Characters](#) because the value concerns a character? No, one chooses [New Histogram For>Trees](#) because the numerous objects being summarized are trees, as the question concerns just one character but many trees. For each tree, what is being calculated is a value that relates to a character.

Contents

- [Selection of objects in charts](#)
- [Auto-recalculation](#)
- [Histograms](#)
 - Example: [Robustness of estimated bias in character evolution](#)
 - Example: [Compositional bias along a sequence](#)
 - [Calculating and formatting options](#)
- [Scattergrams](#)
 - Example: [Canonical Variates Analysis](#)
 - Example: [Correlation between variability and hydrophobicity](#)
 - [Calculating and formatting options](#)

Selection of objects in charts

Most histograms and scattergrams depict the values of objects— characters, taxa, trees — that can be selected. If you select these objects elsewhere in Mesquite, for example by selecting a column (character) in the Character Matrix Editor, then this selected objects will be highlighted in the chart. You can select the objects directly in the chart by clicking and dragging with the arrow cursor.

When objects are selected in the chart, and Copy is selected, then a list of the selected data points is copied to the clipboard. Otherwise if no objects are selected, then Copy puts the list of all data points into the clipboard.

Auto-recalculation

By default, charts are recalculated whenever Mesquite detects that the data or assumptions underlying the chart have changed. If the chart calculation takes a long time, then this can lead to many delays if you need to make many changes in the data or assumptions. You can temporarily turn off the automatic chart recalculation by deselecting the Auto-recalculate menu item in the Histogram or Scattergram menu.

Histograms

The histograms available via the Analysis menu are:

- **Characters** — the items whose values are summarized are characters. These could be all of the characters of a matrix, or a series of simulated or randomized characters.
- **Character Matrices** — the items whose values are summarized are whole character matrices. These could be all of the matrices stored in a file, or a series of simulated or randomized matrices.
- **Taxa** — the items whose values are summarized are taxa within a taxa block.
- **Tree blocks** — the items whose values are summarized are blocks of multiple trees. That is, each item is a block of several trees. These could be all of the tree blocks stored in a file, or a series of simulated or randomized tree blocks.
- **Trees** — the items whose values are summarized are trees. These could be all of the trees within a tree block, or a series of simulated or randomized trees.

Examples

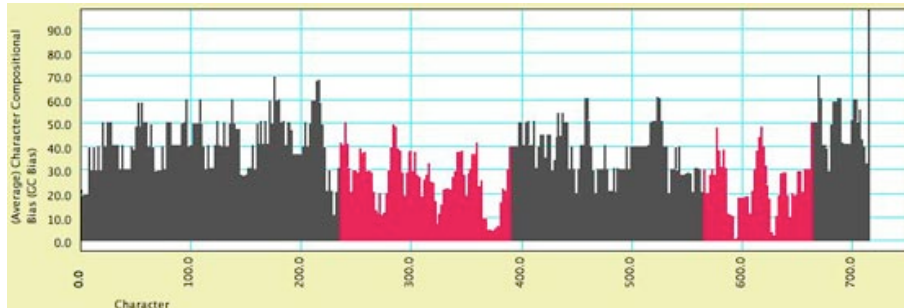
Robustness of estimated bias in character evolution — Suppose a biologist estimated the bias in the rates of gains versus losses in a character's evolution on a given tree. How might the estimate depend on accuracy of the tree's branch lengths? To answer this, one could see how the estimate varies when noise is added to the branch lengths of the given tree. First have a tree window available with the given tree

showing. Then:

- Select the menu item Analysis>New Histogram For>Trees
- A dialog box with heading "Value to calculate for trees" should appear. Choose **Tree Value using character**, because the desired value depends on the tree and a character.
- In the dialog box "Value to calculate for tree with character", ask to show secondary choices then choose **Forward/Backward rates**
- In the dialog box "Source of characters (for Forward/Backward rates)", choose **Stored Characters** (presuming you already have your character of interest entered in a data matrix).
- In the dialog box "Source of trees (Trees chart)", choose **Randomly Modify Current Tree**.
- In the dialog box "Random modifier of tree", choose **Add Noise to branch lengths**
- You will then be asked to indicate the variance of the noise, and the number of trees to chart. You may be asked other questions, depending on whether your data file includes multiple matrices. Then, a histogram should appear to answer your query. (The calculation may take while.)

Compositional bias along a sequence – A biologist with DNA sequence data wants to see how the relative frequencies of A, C, G and T vary along the length of the sequence. To see this:

- Select the menu item Analysis>New Histogram For>Characters
- In the dialog "Value to calculate for characters (for Character Values Chart)", ask to show secondary choices then choose **Character Compositional Bias**
- Choose **Stored Characters** as the source of characters (assuming you have your DNA matrix in the file).
- The chart may initially appear uninteresting, but adjust as follows:
 - Select Histogram>Orientation>Values (Y) by Items (X) to cause the chart to show the characters lined up, in sequence, along the X axis.
 - The chart will probably be set to automatically group into categories along the X axis. Try a moving window analysis by selecting Histogram>Grouping on X>Moving Window..., and indicating the width of the moving window and the offset between adjacent window positions. The defaults are 5 and 1 respectively, but you could also try 10 and 2 to smooth further.



Above is an example of how the chart may appear. Some sections of the chart are red because those characters were indicated as belonging to a distinct [character group](#) or partition. In this chart, the introns (marked in red) have a stronger AT bias.

Calculation and Formatting options

The following menu items can be found in the Histogram menu:

- **Orientation**
 - **Number of Items (Y) by Values (X)** – This is the typical "histogram" where the vertical axis shows how many items have the various values arrayed along the X axis. Thus, the X axis represents the value, the Y axis the number of items.
 - **Value (Y) by Items (X)** – This displays the data with items arrayed in sequence along the X axis, and the Y axis representing the value for each item. This may be appropriate for items, like characters (sites in a DNA sequence), which have a natural ordering to them.
- **Grouping on X** – The X axis may be grouped into categories, such that the values falling within a range on the X axis are summarized in a single bar. This submenu controls any grouping.
 - **Automatic** – Mesquite chooses automatically how to group on the X axis
 - **No grouping** – The X axis is not grouped, and thus each item or object (each character, tree, etc.) is represented by a separate bar.
 - **Fixed number of groups** – The X axis is divided into a specified number of groups.
 - **Fixed Group width** – The X axis is divided into groups of a chosen width.
 - **Moving Window** – The X axis is divided into overlapping groups of a chosen width. This serves to smooth the chart by averaging over adjacent values. There are two parameters to set: the width of the moving window, and its increment. The increment is the offset between the starting edge of adjacent moving window positions.
- **Analysis** – In this submenu could be various analytical tools. Two standard choices are Display Mean and Percentiles..., which display the mean value and tails of the distribution.
- **Show Average For Group** – When the chart is in the orientation Values by Items, and there is grouping along the X axis, then each bar may represent several objects (characters, etc.). This

menu item allows you to choose whether the Y axis should show the sum of the values of those objects, or their average value.

- **Show Individual Points in Text** – When the chart has grouping along the X axis, then the text view of the window by default shows sums or averages of the groups. If instead you want the text view to give all of the values for the original objects shown by the chart, then select this menu item.

Scattergrams

The scattergrams available in the Analysis menu are:

- **Characters** – each point in the plot represents a character. These could be all of the characters of a matrix, or a series of simulated or randomized characters.
- **Taxa** – each point in the plot represents a taxon within a taxa block.
- **Trees** – each point in the plot represents a tree. These could be all of the trees within a tree block, or a series of simulated or randomized trees.
- **Nodes** (Available under New Chart for Tree when a tree window is foremost) – each point in the plot represents a node in the tree of the tree window.

These scattergrams show the values of two variables for the objects of concern. For some scattergrams, a choice is given as to whether the **Same** or **Different** calculations should be shown on the two axes. By "Same" is meant that the same calculation is done but with a different parameter value. For instance, if the scattergram is a Taxa scattergram, it could show the character state in continuous character 1 on the Y axis, and the state in character 2 on the X axis. These represent the same calculation (continuous character state value), differing only in the character used. By "Different" is meant an entirely different calculation, such as the asymmetry of a tree on the Y axis and its likelihood score for a character on the X axis.

Examples

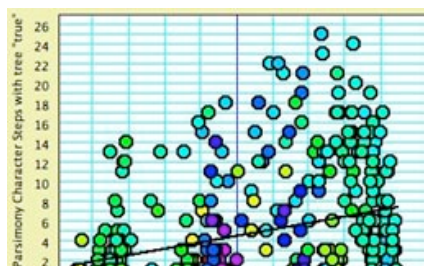
Canonical Variates Analysis – For a sample of specimens measured for a series of variables, how can the measurements be combined to best distinguish predefined groups? Multivariate analyses such as these can be done using modules in the Rhetenor package. Each specimen will be treated as a taxon. A continuous data matrix of the measurements should first be entered, and the taxa be assigned to [groups](#). Next:

- Select Analysis>New Scattergram For>Taxa
- In response to the query about same or different calculations, chose **Same**
- If asked, indicate you want to value for taxa to be **Continuous States of Taxon**. Otherwise, in response to the dialog box "Source of characters (for Continuous States of Taxon)", ask to show secondary choices and choose **"Characters from Ordinations"**.
- In the dialog box "Source of matrices to be ordinated", select **Stored Matrices**.
- In the dialog box "Ordination (for Character Source)", select **Canonical Variates Analysis**. (You may need to ask for secondary choices)

Correlation between variability and hydrophobicity – Do amino acid positions in proteins tend to evolve more quickly or more slowly depending on how hydrophobic they typically are? Mesquite does not yet have direct way to estimate rate for protein characters, but we can approximately compare relative rates by comparing the number of parsimony steps for characters on a tree. First, begin with a file containing a protein data matrix and an open tree window showing a tree. Next:

- Select Analysis>New Scattergram For>Characters
- In the dialog box "Source of characters (For Characters scattergram)" choose **Stored Characters**.
- In response to the query about same or different calculations, chose **Different**
- On the X axis we will put hydrophobicity. Thus, in the dialog box "Values for X axis", ask for secondary choices and choose **Protein Site Property**.
- In the dialog box "Property of Amino Acid" choose **Kyte & Doolittle Hydrophobicity**.
- In the dialog box "Values for Y axis", choose **Character Value with Current Tree**. In the dialog box "Value to calculate for character" chose **Parsimony Character Steps**. Indicate you want **Current Parsimony Models** to be used in the parsimony calculations.

The following scattergram shows the results of such an analysis, with two additions. First, the dots are colored by a third variable, the mean molecular weight of amino acids at that site. This can be done by selecting Color by Third Value from the Color menu, and in the dialog box "Values by which to color spots in the scattergram" asking for secondary choices.



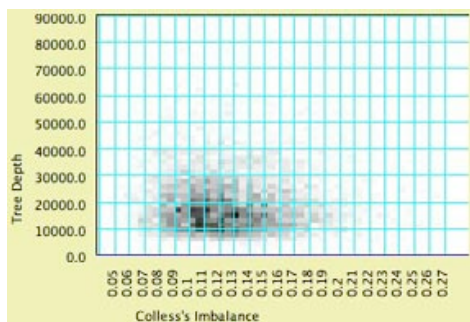


Second, the analysis assistant Scattergram Regression Diagnostics (part of the [PDAP](#) package) is in use, and shows the regression line. The text view of the window shows the details of the analysis. The correlation is highly significant.

Calculating and formatting options

The following menu items can be found in the Scattergram menu:

- **Marker Size** — allows you to choose the size of the dots of the scatterplot
- **Analysis** — In this submenu could be various analytical tools. For instance, if [PDAP](#) is installed, you can request regression and correlation analysis under Scattergram>Analysis>Other Choices.
- **Special Effects**
 - **Show Density** — this shades the background of the chart according to the density of points. To see this, you may want to turn off "Show dots" (see below). Here is an example:



- **Join the Dots, Join Last to First, Thick Joins** — These control whether and how a line is drawn between dots of the scattergram. These options are used to indicate molecular structure as in the cytochrome B example in `Mesquite_Folder/examples/Molecular/06-cytochromeB.nex`
- **Show dots** — determines whether or not the dots representing objects in the scatterplot are drawn individually

In addition, if the scattergram is of characters, a **Color** menu will appear that allows you to color the dots according to a third value of the characters.

Scripting and Macros

Mesquite's scripting system allows modules and other objects to be scripted using text commands. Mesquite uses scripting by text commands as follows:

- When a menu item, button, or tool is used, Mesquite sends a text command to the appropriate object, informing it of the use. This means that most user interface commands can also be executed in scripts and macros.
- When Mesquite re-reads a NEXUS file, it attempts to return windows and analyses to approximately the same condition as when the file had been saved. It does this by reading scripting instructions in a **MESQUITE block** that Mesquite had written into the file when saved. Thus, Mesquite writes scripts called "Snapshots" to later instruct itself.
- Macros are script files which, if placed in the recognized directories, appear in submenus as selectable menu items. When selected, the macro is executed. Mesquite recognizes directories called "macros" within the Mesquite_Folder directory and the "Mesquite_Prefs" subdirectory of the Mesquite_Support_Files directory.
- Not all scripts in MESQUITE blocks need to be written by Mesquite. Users can write MESQUITE block scripts by hand.

The text commands are sent to modules or other objects. The particular text commands to which a module or other object responds will depend on the object. When Mesquite shows a web page for the first time, it attempts to compile automatically documentation for scripting commands that is available. This includes commands predefined by the scripting language, and commands belonging to particular modules. This compiled documentation is available from the web page linked from the "**Scripting Commands**" menu item in the Help menu when Mesquite is running.

Mesquite's scripting language is not as human-friendly as it might be at present, especially in its handling of variables and aspects such as the lack of "else" statements. Since the vast majority of Mesquite scripts won't be written by humans but by Mesquite itself, that is not such a liability as it might seem. We hope to reform the scripting language in the future.

Macros

Scripts performing some special function can be written and distributed by programmers to end users as "Macros". For some calculations or display functions, the number of different analyses a user might like to do are too many to be easily supported by a graphical user interface. For instance, the user might like to print the ancestral state reconstruction of a character using a series of different stepmatrices. One can invent many such scenarios in which something repetitious is needed, and a module would be hard-pressed to maintain each as an option. Thus, small scripts using Mesquite's scripting language can be written and placed in the "macros" folder. These macros will appear as options in an appropriate place, depending upon where they are applicable. When the user selects the menu item, the script is executed.

Macros can also be automatically composed by Mesquite. As noted above, Mesquite composes a script to place in NEXUS files that it saves. This script applies only to the particular saved file. In some contexts, you can ask Mesquite to save a script as a macro file. This is currently available in only a few contexts. Save Window as Macro in the Analysis menu and Window menu save macros that attempt to reproduce the condition (including analyses) of the foremost window. This is useful to reproduce a complex chart with a different data file, for example. Save Macro For Tree Drawing in the Drawing menu of a Tree Window creates a macro by which you can later reproduce the current appearance of the tree drawing (background color, font, tree form, orientation, and so on). Save Tree Analysis As Macro in the Tree menu of a Tree window creates a macro by which to reproduce a particular analysis with the tree. Macros so created are stored in Mesquite_Support_Files/Mesquite_Prefs/macros, and could be (for instance) shared by users.

Learning about scripting

Users can learn about the scripting language from this web page, but also by inspection of existing scripts. Some sources of scripts to examine are:

- The **MESQUITE block** within NEXUS files saved by Mesquite. If you want to know how to script a tree window to trace a character, for instance, open a tree window, turn on character tracing, and save the file. You can look at the file with a text editing program to see how such a script would be written
- If there are any **macros** installed, you can examine the files that specify them. These files are simple text files with scripts. Macros would be found in folders called "macros" within the Mesquite_Folder directory, and in the macros folder of the "Mesquite_Prefs" directory of the Mesquite_Support_Files directory.
- The menu item [Windows>Scripting>Show Snapshot](#) shows a **snapshot** for the window – the script that would be needed to set the window to its current state.

Another way to learn about the language and particular scripting commands is to look at the web pages linked from the "**Scripting Commands**" menu item in the Help menu when Mesquite is running. In

particular, examine the page on universal commands, which shows basic commands available regardless of the object being scripted, including the basic flow control and variable-defining commands of the system. (We cannot provide a link here to this web page because this web page is created by Mesquite when it runs, and is placed in a location that depends on your particular computer. After running Mesquite once, you may want to find some of these files and save a bookmark/alias/shortcut to them – the file on basic scripting commands is named 'puppeteer.html').

Commands within scripts

In general, a command found within a script takes the following form:

```
commandName argument1 argument2 ...;
```

The commandName is a single word (token); the arguments can be multiple tokens, so long as the module knows how to interpret them. Typically each argument is a single token (though this may be string of multiple tokens converted to a single token by quotation).

Scripts

A script consists of a series of commands. At each stage in the script, there is an implicit recipient of the command given (for instance, a module or a window). For instance, here is a script within the MESQUITE block of a NEXUS file. To the right of the commands are comments to explain them.

```
Begin MESQUITE;
  getNumberOfDataSets;      [file coordinator queried for number of data sets]
  Integer.dataSets *it;      [storing number of data sets in integer variable 'dataSets']
  getEmployee 'Data Window Coordinator'; [querying for reference to Data Window Coordinator module]
  tell It;                  [commands to follow will be sent to the Data Window Coordinator]
    Integer.dataNum 0;       [Define integer variable 'dataNum' and assign it 0]
    for *Integer.dataSets;   [for loop; cycle as many times as there are data sets]
      showDataWindow *Integer.dataNum; [commands to make a data window for dataset]
      tell It;              [commands to follow will be sent to module that makes data window]
        showWindow;         [tells the module to make the data window visible]
      endTell;              [finished sending commands to the module that makes the data window]
      increment.dataNum;     [add 1 to the variable 'dataNum']
    endFor;                 [end of the for loop]
  endTell;                 [finished sending commands to the Data Window Coordinator]
  getNumberOfTaxas;         [file coordinator queried for number of sets of taxa]
  Integer.taxaSets *it;      [storing number of sets of taxa in integer variable 'taxaSets']
  getEmployee 'Tree Window Coordinator'; [querying for reference to Tree Window Coordinator module]
  tell It;                  [commands to follow will be sent to the Tree Window Coordinator]
    Integer.taxaNum 0;       [Define integer variable 'taxaNum' and assign it 0]
    for *Integer.taxaSets;   [for loop; cycle as many times as there are sets of taxa]
      makeTreeWindow *Integer.taxaNum; [commands to make a tree window for set of taxa]
      tell It;              [commands to follow will be sent to module that makes tree window]
        getTreeWindow;      [queries the module to return a reference to the tree window itself]
        tell It;            [commands to follow will be sent to the tree window]
          newAssistant 'Trace Character History'; [the tree window is asked to hire a module]
        endTell;           [finished sending commands to the tree window]
        showWindow;        [tells the module to make the tree window visible]
      endTell;             [finished sending commands to the module that makes the tree window]
      increment.taxaNum;    [add 1 to the variable 'taxaNum']
    endFor;               [end of the for loop]
  endTell;               [finished sending commands to the Tree Window Coordinator]
END;
```

This MESQUITE block causes Mesquite to show a data window for each of the data sets, and a tree window for each of the Taxa blocks; the tree windows are shown with a character traced.

This script illustrates some of the features of Mesquite's scripting language:

- There is a **current object** to which commands are sent. Which object is being commanded can be changed by a "tell" command. Initially, the implicit recipient of the commands is the FileCoordinator of the file in question.
- Any given command may **return an object** that is stored within the variable referred to by "It". Thus, in the above script, the makeTreeWindow returns the module hired to supervise the tree window. The immediately following command, "tell It" thus shifts the recipient of commands to this tree window module.
- The scripting language has **variables**. They are defined or their values set by commands beginning, for instance, Integer.myIntegerName or Object.myObjectName. Their values can be utilized by prefixing their names by an asterisk, for instance *Integer.myIntegerName.
- The scripting language has **control flow**, including if, for loops, and while loops.

Some of the lines of this script are commands unique to particular commandable objects (getNumberDataSets, getEmployee, showDataWindow, etc.); others are predefined by the scripting language (e.g., Integer, tell, for). More details on particular commands can be found in the web pages linked from the "Scripting Commands" menu item in the Help menu when Mesquite is running. Some details are found below.

Variables: Integers, Strings and Objects

Three sorts of variables are supported. Reference to each requires the type of variable with name appended, as in "Integer.numberOfCharacters" or "Object.treeDrawCoordinator". The generic variable "it" refers to the object last returned by a command.

When the variable is passed as an argument for a command, it should be preceded by an **asterisk**. This allows the system to know that a variable, and not a constant string, is being passed.

Numerical variables

Two sorts of numerical variables are supported: Integer and Number. The former contain whole numbers. The latter can contain whole or decimal numbers. Reference to each requires the type of variable with name appended, as in "Integer.numberOfCharacters" or "Object.treeDrawCoordinator". The generic variable "it" refers to the object last returned by a command.

When the variable is passed as an argument for a command, it should be preceded by an asterisk. This allows the system to know that a variable, and not a constant string, is being passed. The commands concerning variables are:

- **Integer.[name] [number];**— this declares an integer variable of name "name" and assigns it the value given by the number. If the variable already exists, its value is replaced by the number. The number may be represented by a constant, as in "Integer.counter 5", by an integer variable, as in "Integer.counter *Integer.previousCount", or by a String variable that contains a string that can be parsed into an integer, as in "Integer.counter *String.countString". If the number is indicated as "random", a random integer will be placed in the variable.
- **increment.[name of integer];**— this adds one to the integer's value
- **decrement.[name of integer];**—this subtracts one from the integer's value.
- **Number.[name] [number];**— this declares a numerical variable of name "name" and assigns it the value given by the number. If the variable already exists, its value is replaced by the number. The number may be represented by a constant, as in "Number.rate 0.5", by an integer variable, as in "Number.rate *Number.previousRate", or by a String variable that contains a string that can be parsed into a number, as in "Number.rate *String.rateString". If the number is indicated as "random", a random number between 0 and 1 will be placed in the variable. If the number to be placed into the Number variable is preceded by a '+', the number doesn't replace the existing value of the Number variable, but is added to it (similarly for '-').

String variables

One sort of variable contains a string of text. The command to define and assign values to a string variable is:

- **String.[name] [string];**— this declares a String variable of name "name" and assigns it the value given by the string. If the variable already exists, its value is replaced by the string, unless the string passed to it is preceded by "+" in which case it is appended to the existing string. The string passed may be represented by a literal string, as in "String.name John A. MacDonald", by an String variable, as in "String.name *String.name.firstPM", or by an Object variable, in which case the name of the Object will be used.

Object variables

One sort of variable contains a objects (such as modules, or windows). The command to define and assign values to an object variable is:

- **Object.[name] [reference to object];**— this declares an Object variable and assigns it the object indicated. If the variable already exists, its value is replaced. The reference may be "it", as in "Object.thisModule *It", or an Object variable, as in "Object.thisModule *Object.storedModule".

The variable "It"

Standard Mesquite Commands to modules return an object. In the scripting language, this returned object is stored in the variable "It". Thus, after a command "getNumberDataSets" to the FileCoordinator, the FileCoordinator returns an Integer variable containing the number of data sets. This can be stored in an Integer variable by following the command by "Integer.numDataSets *it". Likewise, "tell" often makes use of "it".

Flow and command control

As noted above Mesquite's scripting language has flow control as well as control of the object to be commanded.

Using "tell" to direct commands

Commands are directed toward commandable objects, including modules, windows, and others. Since different objects might use the same command names, the object to which a command is directed must be indicated. In the scripting language, at any point there is an implicit object being commanded. Subsequent commands are directed to a different object using the "tell" command, which must be balanced by "endTell". At the root level, the FileCoordinator is being commanded.

Flow control

Flow control statements include "if", "for", and "while". Others are available (such as **ifnot**, **stop**, **exitTell**). Details on these can be found via the web page shown by selecting **Scripting Commands** from the **Help** menu while Mesquite is running.

- **if** [integer or integer variable]; ... **endif**; — the statements between if and endif are executed if the integer variable is non-zero
- **for** [integer or integer variable]; ... **endFor**; — the statements between for and endFor are executed as many times as the initial value of the integer variable.
- **while** [integer or integer variable]; ... **endWhile**; — the statements between while and endWhile are executed repeatedly as long as the value of the integer variable is non-zero.

Debugging

There are a number of commands that are useful for debugging. For instance, if the Command "debug" is placed in the block, a debugging mode will be enabled which reports in the console more details about the commands as they are executed. More information about such commands can be found by selecting the "Scripting Commands" menu item under the Help menu when Mesquite is running.

Technical Details

(See the [developer's documentation](#).) The Puppeteer is in charge of defining the basic language as it relates to variables and flow control. For an object to be scriptable, it must be of the Commandable interface. Commands internally in Mesquite are remembered in objects called MesquiteCommands.

Managing modules

Users will be able to use Mesquite more effectively if they become familiar with the concepts of modules and how they interact. An understanding of the modular system will help the user more effectively give commands to Mesquite via [menus](#) and [scripting](#), and interpret the results obtained.

The page on [how Mesquite works](#) introduces modules and how they cooperate. In brief, modules link together to perform calculations and display the results to the user. The linkage of modules is in the form of a tree, with the root module being the "trunk" Mesquite module itself. Which modules are currently active depends on the files the program has read and the calculations the user has requested.

Installed *versus* loaded *versus* running modules

An **installed module** is one available to be used: it is sitting as a file or a series of files on your disk or web server. Mesquite can find these modules and, if appropriate, can get them started up. Among the installed modules might be some that you never use, because you don't happen to need their calculations.

Not all installed modules are available to a user during a Mesquite run. The reason is that, although a module might be physically installed, it might not be **loaded** by Mesquite at startup. An installed module might not be loaded if, for instance, the user has chosen a configuration to control the loaded packages (see Activate/Deactivate Packages in the File menu). The configuration file might indicate that only some of the packages are to be loaded and available.

When Mesquite starts up, or when the user requests new windows, calculations, and so on, some of the loaded modules are started up to do the work. These are the **running modules**.

This characterization is a bit misleading, however, since many active copies of a single module on disk can be running at once. That is, the installed module on disk can be thought of as a blueprint for a machine, and each time Mesquite needs to start up a module, the machine is built and started. If Mesquite needs another machine of the same sort, it can build a new one from the blueprint. Many machines (running modules) from a single blueprint (loaded module) can be built and be running simultaneously.

(To the programmer, the loaded modules are the available classes that are subclasses of the basic module class; an active module is an object instantiated from one of these classes.)

Where modules are installed

To be installed and found by the Mesquite system, a module must be within the **mesquite** directory of **Mesquite_Folder**. Furthermore, the active configuration must specify that the module's package be loaded. By default, all installed packages are loaded.

Thus, if you open up the Mesquite_Folder, you might see the following

- com
- corejava
- docs
- documentation.html
- edu
- examples
- images
- Jama
- JSci
- lesser.txt
- pal
- mesquite. In this directory, you might see:
 - ancstates
 - assoc
 - basic
 - batchArch
 - categ
 - charMatrices
 - charts
 - coalesce
 - configs
 - cont
 - distance
 - genesis
 - io
 - jama
 - jsci
 - lib

- o lists
- o Mesquite.class
- o **minimal**
- o molec
- o ornamental
- o pairwise
- o pal
- o parsimony
- o rhetenor
- o search
- o stochchar
- o treefarm
- o **trees**
- o trunk

Each of the items in bold above is a directory (folder) that contains modules. Within **minimal**, you might see:

- BasicFileCoordinator
- Defaults
- DrawHierarchy
- InterpretNEXUS
- ManageForeignBlocks
- (and so on)

Each of these directories is for one module. For a module to be found and used by Mesquite, it must be present in a directory whose name is the same as the class file for the module (e.g., the class file for the module "Basic File Coordinator", BasicFileCoordinator.class, must be in a directory named **BasicFileCoordinator**).

Modules currently in use

While Mesquite is running you may want to learn what modules are currently running. In the **Help** menu the **Active Module Tree** menu item displays a window that displays the modules currently running. You can also see the modules responsible for any window by touching on the Modules tab in the window's information bar.

If you touch on the name of the module in these module tree displays, a menu will drop down that allows you to find out information about the module.

Information pages and Manuals for modules

Mesquite gathers information about each module and summarizes it in an **information page** for each module. These are linked from the page available via the **Modules Loaded** menu item in the **Help** menu while Mesquite is running. As well, some individual modules have their own **manuals** in the form of web pages. These are also linked from the Modules Loaded web page.

How Mesquite works

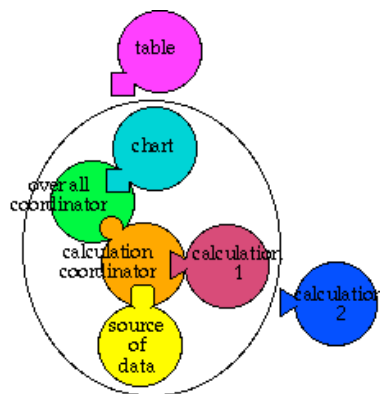
Mesquite operates via the cooperation of modules, each of which performs some function or duty. There is one core module, the **Mesquite trunk module**. When this module starts up, it finds all of the available modules. (Mesquite does this by looking into the subdirectories of the **mesquite** directory.) From each module, it gathers information, including its name and the functions the module promises to perform.

A programmer's account of Mesquite's modular architecture is given in the [developer's documentation](#).

The tree of modules

Modules are "hired" by other modules as employees to perform certain tasks. Thus, the core Mesquite module hires a "File Coordinator" module to manage most activities, including coordinating the reading and analyzing of data files. The File Coordinator itself hires other modules to do the actual file reading and analyses. The file reading module may hire other modules to handle particular parts of the data file, one for the data matrix, another for the trees, and so on. With modules hiring modules hiring modules, the modules in use at any time are linked together in employee-employer relationships that as a whole take the form of a **tree**, with the root module being the core Mesquite trunk module. (To see the tree of modules active at any time, choose the menu item "Active Modules" from Mesquite's Help menu, or by touching the Modules tab in a window's information bar.)

Here is a diagram of a section of the module tree. The five modules in the oval are engaged in calculating and presenting some result to the user. (The two modules outside the oval are patiently waiting as alternatives.) The over-all coordinator module has hired a calculation coordinator module, which in turn uses one module to supply data and another to do the actual calculation. The over-all coordinator also hires a module to display the results (in this case, a charting module).



Kinds of modules: Duties

A module will decide which other modules would be appropriate to hire for a particular task according to the **duties** that the various modules promise to perform. Each module belongs to a particular kind of module, and these kinds amount to a claim as to what duties will be performed. (Technically, these "kinds" are subclasses of the general Mesquite module class, and a particular module can choose which kind it is by what subclass it extends.) For instance, one "kind" of module is a DrawTree module. By belonging to this kind, a module promises that, when given a tree, it will draw it. There may be multiple DrawTree modules available -- one may make square trees ("phenograms"), another diagonal trees ("cladograms"), another circular trees, another may plot the nodes of the tree in a three dimensional space, and so on. The fact that a module is of the DrawTree kind guarantees to any employer module that it will perform a task in a predictable way, and be interchangeable with other modules of the DrawTree kind. If a module needs a tree drawn, it could choose any one of these DrawTree modules to hire. Modules can easily fire an existing employee, and hire a new one, to respond to a user's request to change the calculation or display.

Consequences to the user: Flexibility and choice

This system of modules provides great **flexibility** for the user in designing calculations and output. The user can typically choose which particular module of a kind is used, and thus mix and match modules to construct an analysis. In the example of the illustration, the user might substitute the other calculation module (calculation 2) and the other graphical display module (table) and thus arrive at a different calculation and output. If, in the future, someone writes a new sort of calculation module (calculation 3), it can also be plugged in to the system as a new alternative. It can be used not only in the calculation diagrammed above, but in any other analysis in which a calculation of its basic kind is needed. Likewise, if someone writes a new sort of graphical presentation module, it can also be plugged in. This means that the number of alternative analyses and display modes available to the user rises multiplicatively as more modules of different kinds are available.

The diversity of analyses that can be done with Mesquite is valuable in many respects, not the least of which is that it helps overcome the constraining effect that computer programs tend to have on a user's work. However, it also forces the user to make **choices**: to think about exactly what he or she is doing. This may be an advantage or disadvantage. Of course, one could write an artificial intelligence module that advises the user what to do....

It is not just the diversity of analyses that makes Mesquite unusual. It is the fact that we cannot predict what features it will have, since its features depend on the modules currently installed and loaded. Thus, a simple manual for it is not possible. In a normal program we could say "If you want to trace a character, go to the Analysis menu and choose Trace Character. The branches will be shaded to show the parsimonious ancestral states". With Mesquite, the branches might not be shaded, because the reconstructed states might be displayed in some other way, according to whatever display module is in use. The states might not be reconstructed by parsimony. The Trace Character command might not appear in the Analysis menu, since the Trace Character module might be hired by various other modules, and each of these employer modules might choose a different place (e.g., somewhere other than a Trace menu) to give the user the opportunity to choose analyses. We have attempted to describe standard analysis via the links at the left hand side of this page. Beyond those, however, the user can use the examples files for more information about setting up analyses.

Characters and Character matrices

Characters and their character states are means by which to describe the features of organisms. Mesquite supports characters whose states are **categorical** (discrete and not necessarily ordered) or **continuous**. Special versions of categorical characters exist for DNA, RNA and protein sequence data. For more details specific to those types of characters, see the pages on [molecular](#) and [continuous](#) characters.

Characters can exist within **matrices** that are stored in a data file. Thus, one matrix may store a series of categorical characters to describe phenotypic features. A separate matrix may store continuous characters describing measurements taken from the organisms, while a third matrix may store DNA sequence data, in which each aligned site is treated as a character (Mesquite does not currently handle fully unaligned data). Each matrix may have only a single type of character, but a data file may contain more than one matrix.

Characters can also exist outside of matrices. For instance, characters may be created by [simulations and randomizations](#) and used directly in calculations, without at any point being captured in a matrix and stored in a file.

Creating a character matrix

There are several ways to create a character matrix to be stored in the file. Most simply, you can create a blank (empty) matrix by choosing Characters>New Empty Matrix. In the dialog box that appears, name the character matrix and specify the number of characters. You will also need to choose the sort of data the matrix will contain (standard categorical, DNA (or RNA) sequence data, continuous, or protein sequence data). Normally, you will create an empty matrix if you are about to start entering observations about organisms.

It is also possible to create character matrices that are already filled with character states. For instance, if you want to make a duplicate of an existing character matrix, select Characters>Make New Matrix From>Stored Matrices. If you want to create a matrix from the contents of the clipboard, select Characters>Make New Matrix From>Clipboard. Other choices available under Characters>Make New Matrix From> allow you to make and store matrices resulting from simulations of character evolution, randomizations of existing matrices, or other sources.

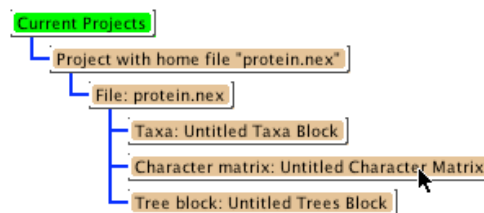
Character matrices can also be read from files, including those in NEXUS and other formats that can be imported.

Deleting and renaming matrices

There are two places you can rename and delete character matrices: in the List of Character Matrices window, and in the Projects and Files window.

In the List of Character Matrices window (available in the Characters menu), you can rename a matrix by editing its name directly. To delete matrices, select the rows corresponding to the matrices to be deleted, and select List>Delete Selected Character Matrices.

To rename a matrix from the Projects and Files window, touch on the "Character Matrix" box:



A drop-down menu will appear with the option to rename the matrix. You can also use this drop-down menu to delete a matrix.

The Character Matrix Editor



Once you have a character matrix, you may edit it using Mesquite's Character Matrix Editor, available at the top of the Characters menu. This is a spreadsheet editor, similar in style to MacClade's. While the Mesquite editor can handle continuous data and has special utilities, for instance to compare matrices, it lacks MacClade's sophisticated features for viewing and alignment of molecular sequence data. Because MacClade and Mesquite share the NEXUS file format, for most data files you will be able to edit matrices in either program to use in the other. Below are instructions as to how to edit a character matrix. Most of the editing can be done in the Character Matrix Editor, but some changes can be made in other windows.

Note that currently most changes you make to a character matrix **cannot be undone!**

The Character Matrix Editor showing morphological data


The Character Matrix Editor showing DNA sequences


Adding, deleting, renaming and sorting taxa and characters

There are several methods for adding taxa or characters to an existing matrix. To add taxa, either choose (Character Matrix) Matrix>Add Taxa... or (Taxa) List>Add Taxa... to add taxa to the end of the matrix, or use the Add Taxa tool () in the character matrix to add taxa at the point in the matrix that is touched. To add characters, either choose (Character Matrix) Matrix>Add Characters... to add characters to the end of the matrix, or use the Add Characters tool () in the character matrix to add characters at the point in the matrix that is touched.

To delete existing taxa, either select the taxa in the Taxa List Window (by touching on the taxon's number at the far left), and choose (Taxa) List>Delete Selected Taxa, or select the entire row for the taxa to be deleted (by touching on the taxon's number at the far left) in the Character Matrix and choose (Character Matrix) Matrix>Delete Selected.

To delete existing characters, either select the characters in the List of Characters Window (by touching on the character's number at the far left), and choose (Characters) List>Delete Selected Characters, or select the entire row for the characters to be deleted (by touching on the characters's number at the far left) in the Character Matrix and choose (Character Matrix) Matrix>Delete Selected.





To rename taxa or characters, choose the I-beam tool () in the Taxa List Window, List of Characters Window, or Character Matrix Window, select the name to be edited, and type the new name.

To change the order of characters, you can select and drag entire characters in the List of Characters Window or the Character Matrix Editor. You can also use the sort tool () to sort characters automatically in alphabetical or numerical order of the column or row on which you touch in these windows.

Entering character data





You can enter character data either one cell at a time, or using tools that allow entry of multiple cells at once. Tools available are shown in the following table.

Tool	Action
------	--------

	I-beam	Selects individual cell and allows you to edit the contents of the cell as you would any standard text.
	Key Touch	If this tool is active, typing a key will cause that single value to be entered into all selected cells. Cells can be selected with this tool; by holding down the Shift or Command keys, multiple cells can be selected.
	Paint Bucket	This tool will quickly fill a block of cells with a particular state. The state ("paint") can be chosen by touching the Eye Dropper on a cell of the appropriate state, or by choosing Set Fill States from the Paint Bucket's drop-down menu.
	Eye Dropper	This tool, when touched on a cell in the matrix, sets the Paint Bucket's "paint" to the states in that cell.

Selecting taxa, characters and cells of the matrix

Mesquite has several tools for selecting taxa, characters, or data cells, as described in the following table.

Tool	Action
	<p>Arrow</p> <p>Selects individual or multiple cells. To add or subtract cells to an existing selection, hold down the Command (or Apple) key as you touch on a cell. To extend a selection to encompass a solid block of cells, hold down the Shift key as you touch on a cell.</p>
	<p>Wand</p> <p>By default, selects all cells possessing the same valued state as the cell touched. That is, if you touch it on a cell with state "1", all cells in the entire matrix with state "1" will be selected. However, using the drop-down menu, you can ask it to choose all cells with a value greater than that touched, or less than. By default, this tool selects cells throughout the entire matrix. Using the drop-down menu, you can ask to restrict the select to a single taxon, or a single character. Holding down the Shift key will add the new cell to the existing selection. Holding down the Command (or Apple) key will add the new cells to the existing selection if you touch on a cell that is not selected, and will remove the cells from the existing selection if the cell is already selected.</p>
	<p>Taxon Wand</p> <p>By default, selects all taxa possessing the same state within the character touched as that in the cell touched. However, using the drop-down menu, you can ask it to choose all taxa with a value greater than that touched, or less than. Holding down the Shift key will add the new taxa to the existing selection. Holding down the Command (or Apple) key will add the new taxa to the existing selection if you touch on a taxon that is not selected, and will remove the taxa from the existing selection if the taxon is already selected.</p>
	<p>Character Wand</p> <p>By default, selects all characters possessing the same state within the character touched as that in the cell touched. However, using the drop-down menu, you can ask it to choose all characters with a value greater than that touched, or less than. Holding down the Shift key will add the new characters to the existing selection. Holding down the Command (or Apple) key will add the new characters to the existing selection if you touch on a character that is not selected, and will remove the characters from the existing selection if the character is already selected.</p>

Copy/paste

You can copy taxon and character names from one region of the matrix to another and from one matrix to another. You can also copy one or more cells in the matrix to the Clipboard, and paste them into another region of the matrix, or into another matrix. Mesquite allows you to do with discontinuous selections, as long as the number of cells selected in the first taxon that is selected while copying is the same as the number of cells selected in the first taxon that is selected while pasting, and the same for subsequent taxa. That is, if you select two cells in taxon 3, one cell in taxon 5, and four cells in taxon 7, and copy this to the Clipboard, then when you paste, you must select two cells in the first taxon in which to paste, one in the next, and four in the last.

Mesquite will not let you paste a block of cells into the matrix while you have selected a differently shaped block in the matrix. However, if you attempt to do that, Mesquite will offer to change the

selection so that covers the same number of cells as in the selection. You may then attempt again to paste.

Editing Names of Characters and States

Character names can be assigned either by editing the column headings in the Character Matrix Editor, or by editing the character names directly in the List of Characters window.

The State Names Editor, available by choosing [\(Character Matrix\) Matrix>Edit State Names](#), allows you to name the states of categorical characters. It will not be available if your matrix is specified as nucleotide or protein data.

Options for viewing the matrix

The Character Matrix Editor allows various display modes. Various display parameters may be set in the [\(Character Matrix\) Matrix>Display](#) submenu. Cells can be colored using the items in the [\(Character Matrix\) Matrix>Color Cells](#) submenu. Cells can be colored by their state, by whether or not a footnote is available, and by other values calculated for each cell. This last facility, "Color By Value", allows moving window analysis to visualize differing GC content or amino acid properties. For instance, with DNA sequence data, select [\(Character Matrix\) Matrix>Color Cells>Color By Value](#). The cells will be colored blue if the site is G or C, white if A or T. By selecting [\(Character Matrix\) Matrix>Moving Window \(for colors\)...](#), you can set the size of the moving window over which GC content is averaged.

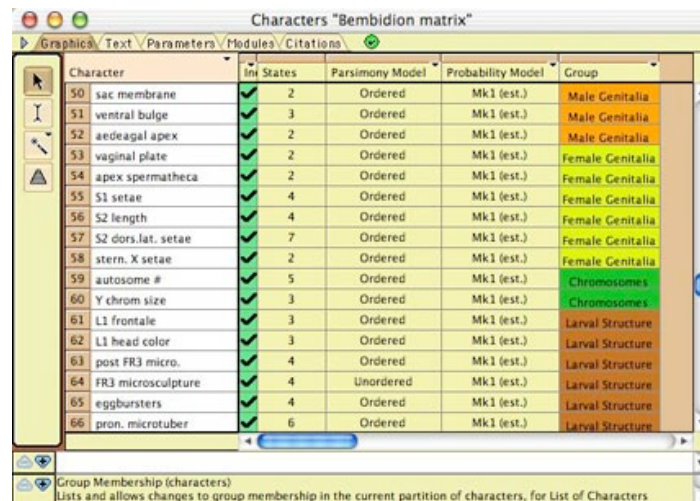
Alterations and Transformations

The following are available in the Alter/Transform menu of the Matrix menu to modify the cells of a matrix:

- Filling selected cells with a specified state: Choose [\(Character Matrix\) Matrix>Alter/Transform>Fill](#)
- Filling selected cells with random states with equal frequency for all states: Choose [\(Character Matrix\) Matrix>Alter/Transform>Random Fill](#)
- Randomly reshuffle the states within a character among the selected taxa: Choose [\(Character Matrix\) Matrix>Alter/Transform>Other Choices...>Shuffle states among taxa](#)
- For nucleotide sequence data, convert the entries in each cell into their complement: Choose [\(Character Matrix\) Matrix>Alter/Transform>Other Choices...>Nucleotide complement](#)
- Reversing a selected molecular sequence: Choose [\(Character Matrix\) Matrix>Alter/Transform>Other Choices...>Reverse Sequence](#)

Changing attributes of a characters

A character, in addition to having states assigned in each of the terminal taxa, may also have other attributes. For instance, a character is marked as included or excluded, and it has assumptions attributed to it, such as a weight and a parsimony model of evolution. These attributes are used in various calculations. They may be assigned in the List of Characters windows, available in the Characters menu.



Character	Ini States	Parsimony Model	Probability Model	Group
50. sac membrane	2	Ordered	Mk1 (est.)	Male Genitalia
51. ventral bulge	3	Ordered	Mk1 (est.)	Male Genitalia
52. aedeagal apex	2	Ordered	Mk1 (est.)	Male Genitalia
53. vaginal plate	2	Ordered	Mk1 (est.)	Female Genitalia
54. apex spermatheca	2	Ordered	Mk1 (est.)	Female Genitalia
55. S1 setae	4	Ordered	Mk1 (est.)	Female Genitalia
56. S2 length	4	Ordered	Mk1 (est.)	Female Genitalia
57. S2 dors.lat. setae	7	Ordered	Mk1 (est.)	Female Genitalia
58. stern. X setae	2	Ordered	Mk1 (est.)	Female Genitalia
59. autosome #	5	Ordered	Mk1 (est.)	Chromosomes
60. Y chrom size	3	Ordered	Mk1 (est.)	Chromosomes
61. L1 frontale	3	Ordered	Mk1 (est.)	Larval Structure
62. L1 head color	3	Ordered	Mk1 (est.)	Larval Structure
63. post FR3 micro.	4	Ordered	Mk1 (est.)	Larval Structure
64. FR3 microsculpture	4	Unordered	Mk1 (est.)	Larval Structure
65. eggbursters	4	Ordered	Mk1 (est.)	Larval Structure
66. pron. microtuber	6	Ordered	Mk1 (est.)	Larval Structure

In the List of Characters window, columns refer to inclusion, parsimony model and probability model (for likelihood calculations). Other columns can be requested for Group Membership, Weight, and (for DNA data) Codon Position. For each of these columns, the assigned attribute can be changed by first either selecting the characters to be changed (if only some characters are to be altered) or selecting the attribute's column (if all characters are to be altered). Then, by touching the name of the column (where an inverted black triangle should appear), a **drop down menu** appears that allows you to make the appropriate specification.

For each of the attributes other than group membership, the bottom three menu items are to store to the file the current specification as a named specification set (like saving a typeset or weightset in MacClade), to replace an existing specification set with the current one, or to load a stored specification set to become the current.

The following are the options specific to each column:

- Inclusion - Include, Exclude, and Reverse allow you to change character inclusion. Reverse changes excluded to included and vice versa. Characters that are excluded don't participate in treelength and many other calculations. Exclusion is not universally respected by the calculations, for some calculations use even characters that are excluded.
- Parsimony model - The Model submenu allows you to select a parsimony model to assign to the characters, for use in parsimony calculations.
- Probability model - The Model submenu allows you to select a probability model to assign to the characters, for use in likelihood calculations and simulations.
- Group Membership - The image above shows Group Membership in the last column. To assign characters to groups, you must first create groups using New Group. For instance, you could create a group Adult Characters and another group Larval Characters. Then, you can assign character to the group using the Set Group submenu. You can also edit the color of the group, and rename the group. The group color is useful to distinguish characters of different groups, for instance in charts or in the Character Matrix editor.
- Weight - With the Set weight menu item you can set the weight assigned to the character. This is used currently on in treelength calculations.
- Codon Positions — This column is available for DNA data. The drop down menu allows you to assign positions.

Charting information about Characters

Informative statistics and values for characters can be viewed or [charted](#) in various windows. In the Analysis menu a **Histogram** can be requested to show the distribution of a value for a series of characters. For instance, the number of parsimony steps in the characters on a current tree can be charted. The **Scattergram** available in the Analysis menu plots characters in a two dimensional space with the X axis being one particular value (e.g., the character's likelihood under one tree), the Y axis another value (e.g. the character's likelihood under a different tree). Values for characters can also be viewed in the **List of Characters** window, where columns can be added (in the List menu) to show selected statistics for each of the characters.

Taxa

Taxa are the fundamental entities in Mesquite: they represent the species or gene copies whose characteristics are recorded in character matrices, whose relationships are summarized in trees. Our use of the word "taxa" isn't the traditional one, which views a taxon as a formally recognized group at any level (e.g., genus, family, order) in a taxonomic classification. Rather, as with other computer programs, Mesquite uses "taxon" as a shorthand for "terminal taxon" (the smallest unit of analysis of relationships, equivalent to "Operational Taxonomic Unit" or "Evolutionary Unit"). Higher level groups are referred to as clades.

Taxa in Mesquite currently must belong to taxa blocks, which are collections of taxa. Thus, the taxon "Homo sapiens" would belong to a taxa block, for instance one called "Mammalia" which may also contain other taxa such as "Mus musculus" and "Ornithorhynchus anatinus".

Creating and managing taxa blocks

To add taxa to a data file, a taxa block must first be created to contain them. Mesquite automatically asks you to create a taxa block when you make a new file. You can later create a new taxa block by selecting [Taxa&Trees>New Block of Taxa...](#) A dialog box will ask you the name of the taxa block and the initial number of taxa (you can add or delete taxa later). You will be then shown the List of Taxa window, in which you can rename the taxa (you can also rename them in the Character Matrix Editor).


Mesquite allows more than one taxa block to exist in a file. Thus, there is a List of Taxa Blocks window which shows you all of the taxa blocks defined. To rename a taxa block, edit its name directly in this window. To delete a taxa block, select its row in this window and choose [List>Delete Selected Taxa Blocks](#). Deleting a taxa block may cause character matrices and tree blocks that depend on it to be deleted.

Managing taxa

You can add and delete taxa in the List of Taxa window and in the Character Matrix editor. In the List of Taxa window, taxa can be added by selecting [List>Add Taxa...](#). In the Character Matrix Editor, taxa can be added either by using the Add Taxa tool or by selecting [Matrix>Add Taxa...](#)

Taxa can be deleted in the List of Taxa window by selecting their rows and choosing [List>Delete Selected Taxa](#). Taxa can be deleted in the Character Matrix Editor by selecting their rows and choosing [Matrix>Delete Selected](#).

Taxa can be renamed by editing their names directly in the List of Taxa window or in the Character Matrix editor, or by using the Name tool in the Tree Window.

To change the order of taxa, you can select and drag entire taxa in the List of Taxa Window or the Character Matrix Editor. You can also use the sort tool () to sort taxa automatically in alphabetical or numerical order of the column on which you touch in these windows.

Assigning group membership

Taxa can be assigned to groups to form a partition of the taxa. Thus, taxa could be assigned to groups according to a traditional taxonomic scheme (some taxa in the group "Vertebrates", others "Invertebrates"), or according to some quality such as distribution ("Neotropical", "Holarctic"). The reason to assign taxa to groups is not to constrain trees or do formal analyses (these groups in most circumstances don't get involved in formal analyses) but rather for reasons of graphics and interface. Groups can be assigned colors, and thus taxa can be highlighted by their group's color wherever they appear (e.g., in charts, in the Tree Window). (One analytical advantage is in multivariate analysis, where Canonical Variates uses the assigned groups as the prior grouping.)

To assign taxa to groups, go to the List of Taxa window. Select [List>Show Column>Group Membership \(Taxa\)](#) to show the column indicating group membership. If you click on the column heading "Group", a menu will drop down with menu items to manage group membership. You must first create groups using New Group. For instance, you could create a group Neotropical and another group Holarctic. Then, you can assign taxa to the group using the Set Group submenu. You can also edit the color of the group, and rename the group.

Phylogenetic trees

Phylogenetic trees represent the branching history of descent linking taxa, whether these taxa are species or genes. In Mesquite, a tree refers to the taxa in a particular [taxa block](#) and once created cannot be transferred to refer a different taxa block. (As explained in the page on [taxa](#), "taxon" here is used as shorthand for "terminal taxon" or "OTU".)

Characteristics of trees

Mesquite typically treats trees as rooted, although it is possible to de-root trees. Trees may contain polytomies (multifurcations) and also unbranched internal nodes. A tree in Mesquite does not need to contain all of the taxa in the taxa block, and indeed can contain as few as one taxon. Unlike MacClade, Mesquite does not support trees with observed taxa fixed in ancestral position. Mesquite can read, edit and write branch lengths in trees. In addition to length, a branch may have various other attributes such as width (e.g. for effective population size in population genetics) and color (for display purposes).

Polytomies in trees are interpreted either as "soft" (uncertainty in resolution) or "hard" (simultaneous divergence). The default interpretation is specified in the Defaults menu of the Startup Window, the Project and Files window, or the Log window. A change in this default applies to all projects and files. Individual trees can be marked as using a specific assumption, thus overriding the default (e.g., by using the Set Polytomy Assumption menu items in the Alter/Transform Tree submenu of the Tree menu of the Tree Window). The appropriate assumption for most phylogenetic studies is "soft", but calculations using soft polytomies are extremely difficult, and most Mesquite calculations yield results only with dichotomous trees and those with hard polytomies.

Analyzing trees

Trees can be visualized in various tree windows, and statistics about them presented in [tree windows](#), the [List of Trees](#) window, and in [charts](#). We will not attempt to summarize all of the options here, which are discussed elsewhere, in particular in the analysis links at left.

Tree comparison methods include the following:

- **Tree to tree similarity measures** — The Shared Partitions module measures the number of partitions between taxa shared by two trees. The separately-available [TSV](#) package includes other measures such as the Robinson-Foulds metric.
- **Consensus trees** — This is not available standardly in Mesquite, but can be added with the TSV package, which includes strict consensus trees.
- **Fits of trees into trees** — The fit of a contained tree (e.g. gene or parasite) into a containing tree (e.g. species or host) can be measured by [Deep Coalescence](#) in the coalesce package.
- **Taxon instability among trees** — This module measures for each taxon how variable are its relationships among a set of trees. Taxa that are particularly unstable, i.e. whose placement is especially variable from tree to tree, score high on this index. Taxon instability calculations are illustrated in the example file at `Mesquite_Folder/examples/Basic_Examples/tree_manipulation/13-instability.nex`

Stored trees and tree blocks

Trees may exist within **tree blocks** that are stored in the data file. A tree block is a collection of trees stored as a unit. A single data file may contain more than one tree block, each containing from one to many trees.

Mesquite calculations can use trees other than those stored in tree blocks in a data file. Most calculations can use trees that are temporarily created using simulations or randomizations specifically for the purpose of the calculation and then discarded. Also, a recently edited tree in the Tree Window might not be stored in the file if the user has not explicitly stored it in the file using "Store Tree" or "Store Tree As". More details on storing trees in tree blocks in the file are given [below](#).

To create a new, empty tree block into which to store trees, select **Taxa&Trees>New Empty Block of Trees...** A new block of trees is automatically created if you ask in the Tree Window to store a tree and no block has yet been created to receive trees.

It is also possible to create tree blocks that are already filled with trees. For instance, if you want to make a duplicate of an existing tree block, select **Taxa&Trees>Make New Trees Block from>Stored Trees**. Other choices under **Taxa&Trees>Make New Trees Block from>** allow you to create tree blocks filled with trees resulting from simulations or randomizations, or other tree sources.

Trees stored in a tree block can be renamed or deleted in the **List of Trees window** (available in the Taxa&Trees menu). To rename, edit the tree's name in the list directly. To delete a tree, select the tree's row in the list window and choose **List>Delete Selected Trees**. Statistics about trees can also be viewed in the List of Trees window by adding the appropriate columns using the List menu.

Tree blocks stored in a file can be renamed or deleted in the List of Tree Blocks window (available in the Taxa&Trees menu). To rename, edit the tree block's name in the list directly. To delete a tree block, select the tree block's row in the list window and choose [List>Delete Selected Tree Blocks](#).

Editing trees: The Basic Tree Window

The Tree Window shows a tree that can be edited and used in analyses. A new Tree Window can be requested by selecting [Taxa&Trees>New Tree Window](#). (Mesquite allows multiple tree windows, and so if you select this menu item a second time, you'll get a new tree window.) Many of the Tree Window's fundamental functions are controlled by its Tree menu. This menu contains items to select the tree source, store trees, and alter the tree. A second important menu is the Drawing menu, which controls the appearance of the tree. Additional menu items related to the Tree Window are found in the Analysis menu.

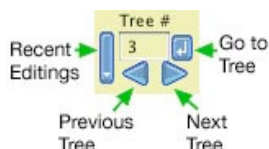
Tree source

The Tree Window shows trees from a particular source (although the tree being shown could differ from those in the tree source, if you've recently modified the tree using the tools). The source of trees might be a tree block in the data file. In this case, as you used the blue arrows at the upper portion of the tool palette, you would be browsing through the stored trees, scrolling from one stored tree to another. Alternatively, the source of trees might be a coalescent simulation, in which case you would be browsing through gene trees representing replicate simulations of the coalescent process. Other tree sources may be available, and can be selected when you request the Tree Window for the first time or by using the submenu [Tree>Tree Source>](#). One tree source, "Default Trees", offers a simple pectinate tree ("Default Ladder"), a full polytomy ("Default Bush"), and a symmetrical tree. It is available merely as a last resort, in case other tree sources are unavailable.

We expect a common confusion will be that users will be unable to find the trees that they recently stored in the file. For example, when a new file is created, there are no stored trees, and hence the Tree Window would be forced to use another tree source (for example, "Default Trees"). If a user then stores a tree in the file (see next section), he or she might hit the blue scroll arrows of the Tree Window expecting to browse the stored trees, only to be shown one of the Default Trees. The problem is that the Tree Window is using Default Trees as the tree source, not Stored Trees. To see the trees stored in the file, select [Tree>Tree Source>Stored Trees](#) to change the tree source to Stored Trees.

Moving from Tree to Tree

To select which tree to view, use the tree scroll at the upper left of the Tree Window:



The left and right blue arrows take you to the previous and next trees, respectively, in the tree source. The Go To button takes you to the tree whose number is entered in the text area. The Recent Editings button offers you a drop down menu by which you can return to recently edited trees. The menu lists only trees that you have edited; it does not list the trees that came directly from the tree source, and is therefore not a complete history of recent trees. The number of little white triangles in the button indicates the number of recently edited trees stored.

Storing trees

Although the Tree Window can be used merely to browse existing trees, it will commonly be used as a tree editor, allowing the user to build a tree according to some prior hypothesis, or to modify trees to explore the effect of changes in the tree.

Once a tree has been edited, the user may want to store this new tree in the file. Exactly how that is done depends on the tree source being used by the Tree Window:

- If the tree source is Stored Trees, then two menu items are available in the Tree menu to store trees, "Store Tree" and "Store Tree As...". For instance, imagine that you scroll through the stored trees until you get to tree number 5. If you edit it using the tools of the tool palette, then you select Store Tree, your newly modified tree will replace the original tree number 5 in the tree block. If instead you choose Store Tree As, you will add your modified tree as a new tree at the end of the tree block, leaving the original tree number 5 untouched. Thus, Mesquite assumes that you are potentially editing the original tree in the tree block whenever you edit the tree. However, the original copy is not replaced by your edited version until you select "Store Tree". Until you select this, your modified tree is maintained as a temporary tree associated with the tree window, and not stored in the tree block.
- If the tree source is not Stored Trees then you can't modify the original trees, because they are

merely temporary trees, produced by something like a simulation. Thus, in this case there is no Store Tree menu item available. Instead, there is a "Store Copy of Tree As" menu item. This acts more or less like Store Tree As, in that the tree is added to a tree block.

You can tell that a tree has been edited to be different from that in the tree source when a black diamond appears in the message area at the lower left side of the Tree Window. The message area turns green when the tree is an unsaved, edited tree and the tree source is not Stored Trees.








Tools

At the left side of the Tree Window is a **tool palette**, containing tools that you can use to interact with the tree. Exactly which tools are available will depend on the modules installed and loaded. In the description of tools below, it will be assumed that a basic set of modules is installed and loaded.





Some tools act when they are touched on a branch of the tree; others act when a branch is touched then dragged and dropped. Some tools behave differently if a key such as shift or control is held down when the tool is used. These details are explained below. Some explanation of the tool is also given in the window's Explanation Area when the tool is selected in the palette. The currently selected tool has its button darker than the rest in the tool palette. If you are running Java version 1.2 or above (e.g., on Linux, Windows or Mac OS X) then the cursor will change to reflect the tool when it is over the tree. If you were running Java version 1.1.8 or below (e.g. on Mac OS 9) then the cursor will remain an arrow.

Some tools have options that can be set. If so, then the button for the tool in the tool palette will have a small black triangle indicating the availability of a drop down menu. If you touch the button and hold down the mouse for a moment, the menu will appear to allow you to make choices.

The following tools **change the topology** of the tree (the fundamental relationships among taxa implied by the tree). Some of these tools might not be available if their controlling modules are not installed or loaded.













Tool	Action
 Arrow	If touched on a branch, dragged and dropped on another branch, the former branch is reattached to the latter. Also can be used to select taxa.
 Interchange branches	If touched on a branch, dragged and dropped on another branch, the two branches exchange positions. Can be used to rearrange a polytomy, but can also be used for branches distant on the tree.
 Collapse branch	Collapses branches to yield polytomy. Also removes unbranched internal nodes.
 Collapse all	Collapses all branches in clade. If option or ALT is held down, collapses all branches below node touched.
 Reroot	Reroots tree at branch touched.
 Prune clade	Cuts clade out of tree. Taxa cut out remain within the data file, but are not included in this particular tree.
 Insert unbranched node	Inserts node along branch. Currently used primarily in combination with assigned lineage widths to indicate population fluctuations for coalescent simulations.

The following tools change the **branch lengths or widths** of the tree.

Tool	Action
 Assign branch length	Brings up a small editable area in which to enter a branch length.
 Stretch branch	Click and drag on branch to stretch its length.
 Adjust node position	Stretches branch lengths above and below node to allow node to change position without changing positions of other nodes.
 Assign lineage width	Assigns a width to the branch. Currently used to assign effective population sizes for population genetics calculations.

The following tools affect the **appearance** of the tree, or change attributes of its branches, but do not change the topology or branch lengths. Thus the changes these tools make will not affect most calculations.

Tool	Action
------	--------

	Ladderize	Rotates nodes to bring the largest clades on the right side of the tree. Does not affect topology of tree. If option or ALT key held down, ladderizes to left.
	Draw clade as triangle	Draws clade as small triangle to save space in drawing the tree. The full relationships within the clade remain, but are hidden. Clicking again will expand the clade to be viewed again. This drawing mode may have bugs.
	Magnify clade	Magnifies the image of the tree so that only the clade touched is shown. Touching again on the node shows the whole tree
	Branch info	Shows available information about the branch and calculations concerning it.
	Color branch	Colors the branch. Control-click colors clade. Repeat click erases. Shift-click shrink wraps color. Touch and hold the button to obtain a menu to select color used. This is not equivalent to the Fix States tool of MacClade.
	Name node	Gives a name to the node. This name must be unique, and different from the names of terminal taxa.
	Select branch	Selects a branch. Control-click extends selection. Shift-click shrink wraps selection.
	Select branches in clade	Selects all the branches in a clade. Control-click extends selection. Shift-click shrink wraps selection.
	Select taxa in clade	Selects all the taxa in a clade. Control-click extends selection. Shift-click shrink wraps selection.
	Annotate node	Attaches a note to a node. An asterisk appears over the node. The note can be seen by passing the Annotate cursor over the branch, or by turning on "Show Notes On Tree" using the menu that drops down from the Annotate node button in the tool palette.
	Show picture	Shows image attached to taxon. Click again to hide. Control-click to attach picture.
	Hyperlink	Shows web page or another data file. Shift click to enter link explicitly. Control-click to select local file.

Alter/Transform menu items

In the Tree menu of the Tree Window are two submenus by which you can change the tree, "Alter/Transform Tree" and "Alter/Transform Branch Lengths". These provide various utilities to change the tree, including its branch lengths. Some, such as "Scale All Branch Lengths", affect all branches of the tree simultaneously.

Analyses

The Tree Window has, in its **Analysis** menu, menu items that yield analyses using the current tree in the Tree Window. Which analyses are available depends on what modules are installed and loaded. Typically the following will be available at least:

- Trace Character History - Illustrates a history of character evolution on the tree. Described in detail in the chapter on [reconstructing ancestral states](#).
- Tree Legend - adds a small legend to the Tree Window. The legend can describe information about the tree, including the results of calculations done using the tree. When the legend is in the Tree Window, an additional menu appears, the Legend menu, by which you can choose to show or hide lines of information.
- Values at Nodes - shows values associated with nodes, either by coloring the nodes or labeling them. These values describe information about the nodes, such as a statistic calculated over the clade above the node. At the moment there are few such values available (notably, the reconstructed character state in a continuous character, also available through Trace Character History).

How trees are drawn

There are many options for the appearance of trees as drawn in the Tree Window and other windows displaying trees. Trees may have diagonal, square or circular branches; they may be drawn black on white or green on blue; they may be drawn so that apparent branch length is proportional to assigned branch length. These options are controlled by the Drawing menu.

Some of the menu items in the Drawing menu are:

- Tree Form - Chooses general form of the tree. Some are merely graphical, others involve analyses such as PlotTree and PlotTree3D (which plot the tree in a 2D or 3D space) and Contained Associates (which shows gene trees within species trees, or parasites within hosts).
- Set Current Form As Default - Sets as default the tree drawing style currently in use (e.g. Diagonal Tree, Balls & Sticks etc.). It does not capture all of the current parameters chosen for that drawing form (background color, spot sizes, etc.).
- Background Color - Sets the color of the field on which the tree is shown.
- Branch Color - Sets the color of the branches. These color are not retained with the tree, but are for graphical purposes only. In this regard it is different from the Color Branch tool, whose colors are retained with the tree.
- Size to Window (Only in the basic Tree Window) - Shrinks the tree to fit into the window. If this is turned off, then the tree exists in a panel that may be much bigger than the tree window itself. To see various parts of this panel, you can use the scroll bars or the Bird's Eye View box that appears at the lower left corner of the Tree Window.
- Drawing Size (Only in the basic Tree Window) - When Size to Window is turned off, this item allows you to set the size of the panel in which the tree is drawn.
- Float Legends (Only in the basic Tree Window) - When Size to Window is turned off, this item allows you to choose whether the legends such as the Tree Legend and Trace Character Legend are anchored to the panel in which the tree is drawn or, anchored to the window itself.
- Font, Font Size, Font Color - Sets the font characteristics of the taxon names and node names.
- Names - Determines what names are shown and whether taxon names are colored by the [taxon group](#).
- Orientation (Only for some tree forms) - Chooses whether the root to terminal orientation is up (classic phylogeneticist or paleontologist), to the right (molecular evolutionist), bottom (population geneticist or mathematician) or left (?)
- Branches Proportional to Length - Determines if the drawn lengths of branches are proportional to their assigned lengths. Otherwise, the tree drawing routine chooses node positions at its own convenience.
- Save Macro for Tree Drawing - Saves the current tree drawing specifications (form, color, orientation, etc.) as a macro. The macro will then appear under the Macros For Tree Drawing submenu, and can be used later to set other tree windows to use the same specifications.

Printing trees and saving graphics files

Mesquite's Tree Window offers two menu items for printing trees: Print Tree and Print Tree to Fit Page. The former prints the tree in its current size, even if that requires it to be placed over multiple pages. The latter automatically reduces or enlarges to tree image to fit a single printed page. It attempts to choose landscape or portrait mode to maximize the size of the fitted image.

There is as yet no direct way to save graphics files of tree images, but satisfactory results can be obtained if you ask to print the tree image, then you use your operating system's Print dialog box to direct the output to a file (e.g., postscript or pdf) instead of a printer. These files may be editable by programs such as Adobe Illustrator.

Dependent Tree Window

This window shows the same tree as in the Tree Window (hence it is "dependent"). It is available in the **Tree** menu of the Tree Window. It is useful when analyses or graphics are desired for the Tree Window, but which would conflict visually with currently running analyses or graphics. Thus, the Dependent Tree Window gives additional space on which to display graphics and analyses. As the tree in the Tree Window is changed, the tree in the Dependent Tree Window is also changed. Note: If the Tree Window is closed, the Dependent Tree Window will close also.

Users may find the menus of the Tree Window and Dependent Tree Window somewhat confusing when the latter is in use, as both windows include some menus of the other. This is a consequence of Mesquite's automatic menu arrangement; we realize it is not optimal in this case.

Mirror Tree Window

This window, like the Dependent Tree Window, shows the same tree as in the Tree Window. However, the Mirror Tree Window shows it twice, in mirror image. This allows you to compare, for instance, two different traced characters visually, as in [this example](#). The analysis shown on the left side is controlled by the Left side submenu; that on the right by the Right side submenu.

Users may find menus confusing when this window is in use; see comments under Dependent Tree Window.

Multi Tree Window

This window, available in the **Taxa & Trees** menu, shows multiple trees simultaneously. It is not dependent on a Tree Window, but gets its trees from an available tree source.

Basic concepts and vocabulary

Mesquite was designed to explore the evolution of organisms and their characteristics. Here is some basic vocabulary that relates the biology to the implementation in Mesquite.

Taxa

- **Taxon** - One of the basic entities being studied, usually a species or a gene sequence. In Mesquite (as in other programs), the word "Taxon" has shifted from its traditional meaning to become more or less equivalent to "terminal taxon" (the smallest unit of analysis of relationships).
- **Block of taxa** - A collection of taxa grouped together for analysis. In most programs dealing with phylogeny, there is only a single block of taxa allowed in a file (e.g., "Primates", consisting of 20 species of primates). In Mesquite, multiple blocks are allowed (perhaps to study associations among them, in the style of Rod Page's programs), and so it is useful to refer to different blocks of taxa. A block of taxa is equivalent to a TAXA block in a NEXUS file.
- **Group Membership (Taxa partition)** - A subdivision of a taxa block into disjoint subsets. Thus, a taxa block for Mammals may have its taxa partitioned into one of three groups, Carnivore, Omnivore and Herbivore. A taxa partition is equivalent to a TAXAPARTITION in a NEXUS file.

Characters

- **Character** - A variable or characteristic measured or observed for a block of taxa. It could be from molecular data ("Nucleotide at position 367 of the 18S gene"), or it could be phenotypic ("length of wing"). Characters in Mesquite typically exist only within character matrices.
- **Character model** - a model of the evolution of a character. This could be a model for use in parsimony calculations (e.g., "unordered", a cost matrix) or in likelihood calculations or stochastic simulations (e.g., HKY85, GTR).
- **Character matrix** - A matrix representing the character states of various characters for a block of taxa. Each character matrix in Mesquite must be homogeneous with respect to type of character (e.g. nucleotide sequences can't be intermingled with morphometrics data).
- **Character partition** - A subdivision of a characters into disjoint subsets. Thus, a character matrix for insects may have its characters partitioned into one of three groups, larval, pupal, and adult. A character partition is equivalent to a CHARPARTITION in a NEXUS file.
- **Character inclusion set** - A specification of what characters are to be included and what excluded from analyses. This is equivalent (in inverse form) to an EXSET in a NEXUS file.
- **Character weight set** - A specification of what weights are to be applied to characters in summed calculations (especially for parsimony calculations). This is equivalent to a WTSET in a NEXUS file.
- **Parsimony model set** - A specification of what parsimony model of character evolution (i.e., what "transformation type") is to be applied to each character of a matrix. This is equivalent to a TYPESET in a NEXUS file.
- **Probability model set** - A specification of what probabilistic model of character evolution is to be applied to each character of a matrix.

Trees

- **Tree** - A tree representing the relationships of the taxa. If the taxa are species, the tree represents phylogeny; if the taxa are gene sequences, the tree represents a gene genealogy.
 - **Block of trees** - A collection of trees. This might be a set of trees saved by the user, or output by a program like PAUP*.
-

Character Evolution

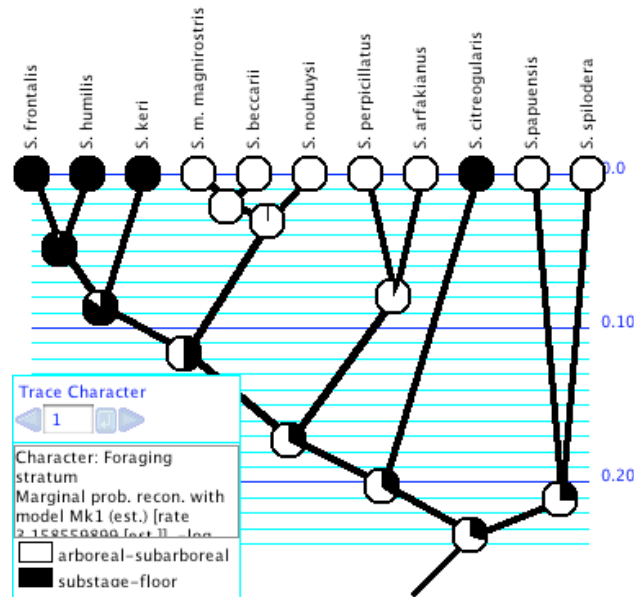
The evolution of characters (measurements, observations) among taxa (species, genes) can be studied both to infer history and interpret processes of change.

- [History: Reconstruction of ancestral state](#)
 - [Processes of character evolution: parameters and correlations](#)
-

Studying the History of Character Evolution

With a phylogenetic tree and a distribution of character states in the observed (terminal) taxa, Mesquite can attempt to reconstruct the character states at ancestral nodes. Two separate issues to consider are the method by which the reconstruction is done, and how its results are displayed to the user. Mesquite currently can use either parsimony or likelihood to reconstruct ancestral states, and has several display methods, including "Trace Character History" which paints the branches of the tree to show the reconstruction.

We recommend highly that you examine the **example files** provided in the folder "Ancestral State Examples". The minimal configuration to use with these examples is "Ancestral States" (indicate this configuration under File>Activate/Deactivate Packages>Choose Configuration), but you can also leave Mesquite in its default All Installed Modules mode.



Trace Character History

The Trace Character History facility graphically represents a history of character evolution on the tree. It is available under the Analysis menu of a tree window (e.g., the basic Tree Window, Dependent Tree Window, Mirror Tree Window, Multitree Window). If you select this you will probably be asked for a source of characters (e.g., stored characters) and a reconstruction method (e.g., parsimony). The tree will be painted to show ancestral states, and a trace legend will appear. The trace legend contains an important text area that gives details of the current ancestral state tracing. You can also see details of the reconstruction by switching the window to Text mode using the tabs at its top.

The Trace menu gives menu items to control the character history and its display. Some important ones are:

Character history source: The character history displayed can be reconstructed using observed states in terminal taxa ("Reconstruct Ancestral States") or can be a simulated history ("Simulate Ancestral States"). The reconstructed states need not be based on actual data, but could be based on simulated data. "Simulate Ancestral States" shows the "actual" history of character evolution branch by branch as it occurred in the [simulation](#), not as it was reconstructed, and thus may show ancestral states that would be unreconstructable, obliterated by subsequent changes.

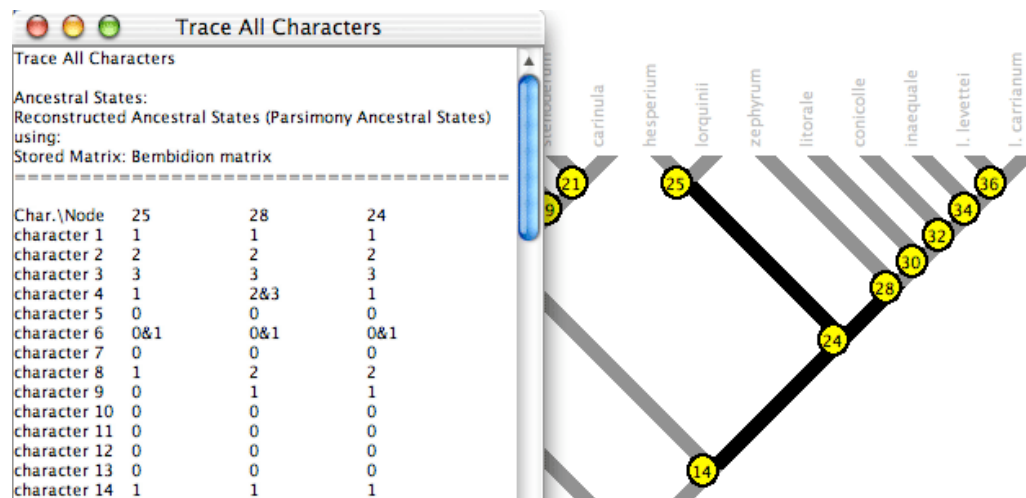
Next, Previous, Choose Character History: Usually, this will allow you to choose which character to view. You can also scroll through characters using the blue arrows in the trace legend.

Trace Display Mode: With Trace>Trace Display Mode>Shade States and Trace>Trace Display Mode>Label States you choose whether to see the branches painted, or have the states indicated in labels. If painted, you can also ask that states be indicated by labels by choosing Trace>Label States.

You can also see details of the reconstruction at a node by holding the cursor over the branch. A description of the reconstructed states will appear at the bottom part of the Trace Character Legend. Another method is to use the Text view of the window (touch on the Text tab at the top of the tree window) and scroll down — a text version of the trace should appear.

Trace All Characters

Trace All Characters summarizes ancestral state reconstructions of many characters simultaneously. To request it, choose **Choose (Tree Window)Analysis>Trace All Characters**. A text window like that shown below will appear, listing the ancestral states reconstructed at each node for each character. Node numbers show up in red on the tree. (Alternatively, spots showing node numbers in the figure below can be turned on in the Tree Window's Drawing menu by selecting Show Node Numbers.)



By default only the selected nodes are listed. (Nodes can be selected using tools in the Tree Window.) You can request to show all nodes by turning off Show Selected Nodes Only in the Trace_All menu. By default all characters are listed; this can be changed using the Show Selected Characters Only menu item.

The ancestral state reconstruction can be controlled in the Trace_All menu of the tree window.

The table is either listed by characters or by nodes; you can switch from one to the other using the Rows are Characters menu item

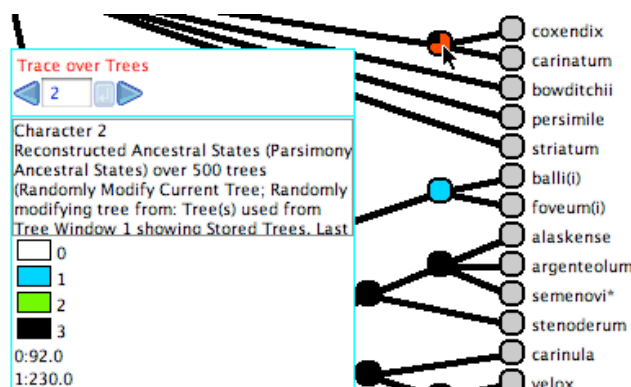
Columns in the table in the text window may not appear perfectly aligned, but it is presented as a tab-delimited table, so you should be able to copy the text and paste it in to a text file to read in to your favorite spreadsheet program.

Trace Character Over Trees

The Trace Character Over Trees facility summarizes ancestral state reconstructions over a series of trees. This is useful to understand how ancestral state reconstructions vary over a series of trees, for instance if there is uncertainty in the tree. It works for **categorical characters** only. Also, Trace Character Over Trees **does NOT calculate a consensus tree** for you. As with all other analyses in the Tree Window, it works with the tree that is given to it by the Tree Window. If you want to make your summary on a consensus tree, then you need to put the consensus tree into the Tree Window first and then request Trace Character Over Trees.

Choose **(Tree Window)Analysis>Trace Character Over Trees**. This examines a series of trees, and for each examines a character's ancestral states on that tree. For each node in the tree in the tree window, it attempts to summarize what ancestral states are reconstructed for that same clade in the series of trees (as long as the same clade exists in the other trees). For example, imagine the tree in the tree window includes the clade Tetrapoda. Each of the series of trees is examined, and if that tree includes the clade Tetrapoda, then its reconstructed ancestral states are examined. If the tree doesn't include Tetrapoda, then it is ignored for the sake of summarizing the tetrapod ancestral states. The tree in the tree window is then decorated to summarize what ancestral states are reconstructed for each of the clades.

Here is an example of Trace Character Over Trees in action:



The cursor is over a node, and thus the legend shows a summary. The node (i.e., the clade it represents) is present in all 500 of the trees examined. In 92 of those trees, state 0 is reconstructed at the node. In 230 trees, state 1 is reconstructed. These two numbers do not add to 500 because some trees show an ambiguous (equivocal) ancestral state reconstruction at the node, and the option to count only the Uniquely Best States is selected in the Count submenu of the Trace_Over_Trees menu. With this option a tree is counted as having a state at a node only if the state is the only optimal state. What is considered optimal depends on the reconstruction method. With parsimony, states are considered equally optimal if they are equally parsimonious. With likelihood, the Decision Threshold is used to decide whether states are good enough to be considered within the optimal set. The alternative to "Uniquely Best State" is "Equivocal". With the Equivocal option for counting, a tree is counted as having a state at a node if the state is within the optimal set, whether or not there are other states within the optimal set. When the Equivocal option is used, the sum of tree counts for the states at a node can more than the total number of trees with the clade, for a tree can get counted multiply at a node, under each state in an equivocal assignment.

An important option is what trees to examine. If the tree in the tree window is a consensus tree, then the trees examined might be the original set of most parsimonious trees that built the consensus. Trace Character Over Trees could then show how the ancestral state reconstruction varies among the most parsimonious trees. The trees examined might also be derived from a Bayesian analysis, and the ancestral states obtained by likelihood, to do an analysis in Lutzoni's style. The trees might be random resolutions of an unresolved tree, or trees with random noise added to branch lengths, and so on. This would allow you to see how the results would vary if the tree changed. See the submenu [Trace_Over_Trees>Tree Source](#) for options.

Parsimony Reconstruction Methods

Parsimony reconstruction methods find the ancestral states that minimize the number of steps of character change given the tree and observed character distribution. They can use different assumptions (models of evolution). For **categorical** characters, the **unordered** states assumption is that one step is counted for any change. The **ordered** states assumption is that the number of steps from state i to state j is $|i-j|$. Thus, the number of steps from state 2 to state 5 is 3 steps. A **stepmatrix** explicitly specifies the number of steps from state to state by a matrix. Mesquite does not yet do parsimony calculations for **irreversible**, **Dollo** and **character state tree** assumptions, although these models are listed in menus and you can assign them to particular characters. For **continuous** characters, the **linear** cost assumption is that the cost of a change from state x to state y is $|x-y|$. The **squared** change assumption is that the cost of a change from state x to state y is $(x-y)^2$.

Mesquite's parsimony calculations attempt to match MacClade's. Some differences remain in special cases of polymorphic terminal taxa with stepmatrices. Mesquite allows hard polytomies in the tree when stepmatrices are used.

Assigning a parsimony model: The parsimony model used for a character's calculations is the model assigned to it, if the character is one stored in a matrix in a file. A parsimony model can be assigned in the List of Characters window. Select the row(s) corresponding to the desired character(s), and then touch on the column heading "Parsimony Model". A drop-down menu contains a submenu that allows you to select the models to apply. You can also change the parsimony model assigned to the character being traced in Trace Character History using the Parsimony Model submenu of the Trace menu. (Recall that Mesquite cannot yet do calculations with irreversible and Dollo models.)

If the characters used in parsimony reconstruction are not stored in a matrix but rather come directly from another source of characters such as simulations, a single parsimony model can be chosen to be applied to all of the characters coming from this source. Thus, for instance, when using Trace Character History, the Parsimony Model submenu of the Trace menu can be used to assign the model to be used.

Creating and editing stepmatrices: To create a stepmatrix, select [Characters>New Character Model>Stepmatrix](#). A window will appear in which you can edit the cost of i to j transitions. The number of states allowed is initially 10 (0 through 9), but you can change the number of states under [\(Edit Stepmatrix\)>Step_matrix>Set maximum state](#). The maximum number of states for a categorical character is 56; the maximum state value is therefore 55. This stepmatrix editor does not do triangle inequality checking (see discussion in manual of MacClade, which does check the triangle inequality).

The parsimony calculations are used also for Treelength and Character Steps.

Likelihood Reconstruction Methods

Likelihood reconstruction methods find the ancestral states that maximize the probability the observed states would evolve under a stochastic model of evolution (Schluter et al., 1997; Pagel, 1999). The likelihood reconstruction finds, for each node, the state assignment that maximizes the probability of arriving at the observed states in the terminal taxa, given the model of evolution, and allowing the states at all other nodes to vary. (In fact, this considers all possible assignments to the other ancestral states.) This is equivalent to the marginal reconstruction of Swofford's PAUP*, or the Fossil Likelihood

reconstruction of Pagel's Discrete.

You can use likelihood for the reconstruction by selecting "Likelihood Ancestral States" when first requesting Trace Character History, or from the Method submenu of the Trace menu after Trace Character History is already active. When using Likelihood Ancestral States in Trace Character History, it is recommended that you use a Tree Form for the drawing that uses spots at the nodes (for example, [\(Tree Window\)Drawing>Tree Form>Balls & Sticks](#)). These spots at the nodes will indicate relative likelihoods with pie diagrams as in Schluter et al. 1997.

At present **only categorical** characters are supported by the likelihood calculations. Models for DNA and protein evolution are not yet available for use by likelihood. Two models of evolution are currently supported, the Mk1 model and the AsymmMk model.

- **Mk1 model** ("Markov k-state 1 parameter model") is a k-state generalization of the Jukes-Cantor model, and corresponds to Lewis's (2001) Mk model. The single parameter is the rate of change. Any particular change (from state 0 to 1 or state 3 to 2, for example) is equally probable. Mesquite's rate of change parameter is equivalent to the q values of Pagel's Multistate program when the q's are constrained to be equal. Thus for a character with three states 0, 1 and 2, the Mk1 model would have an instantaneous rate matrix of the following form:

To	0	1	2
From 0	-	q	q
1	q	-	q
2	q	q	-

- **AsymmMk model** ("Asymmetrical Markov k-state 2 parameter model") has two parameters: one for the rate of increase in state (from 0 to 1, 1 to 2, etc.; the "forward" rate) and one for the rate of decrease in state (from 2 to 1, 1 to 0, etc.; the "backward" rate). For more than two states, this seems a biologically unlikely model, but for two states it provides a simple model that allows a bias in gains versus losses. Mesquite supports two alternative ways to describe the model. The two parameters can be forward rate and backward rate, or overall rate and bias of gains versus losses. Thus, if forward and backward rates are both 0.5, then this can alternatively be described as a rate of 0.5 and a bias of 1.0 (i.e., unbiased). The conversion between the two representations is done by the following formulas: forward rate = overall rate * square root(bias); backward rate = overall rate / square root(bias). This conversion means that forward rate / backward rate = bias. Thus an AsymmMk model would have an instantaneous rate matrix of the following form for a 3-state character:

To	0	1	2
From 0	-	f	f
1	b	-	f
2	b	b	-

This may be an unrealistic model for 3 or more states (why should all increases in state number have the same rate and decreases a different rate?), and so we expect it will be used mostly for two-state characters.

Many programs bundle the rate of evolution into the branch lengths of the tree itself. Thus, to change the rate of evolution, the tree needs to be stretched or shrunk; there is no separate rate parameter that belongs to the stochastic model of evolution. This works well as long as the branch lengths are understood in the same way by the model and the tree, i.e., the tree's time units (calibration of time scale) are the same as that of the model. However, in Mesquite different calculations might make different assumptions about the time scale: coalescence calculations might need the tree's branches measured in generations, while a Jukes Cantor model might assume they are in expected nucleotide substitutions. Thus, many stochastic models in Mesquite have an extra parameter compared to other programs: the scaling of the model to the tree. For this reason Mk1 has a rate parameter to scale the rate against the tree.

If parameters of a model are unspecified, Mesquite currently [estimates](#) them based on the data. **Note:** Mesquite currently estimates parameters on each character separately, not on the entire data matrix. In addition Mesquite's likelihood calculations do NOT estimate branch lengths. They use pre-existing branch lengths (if a branch length is unassigned, it is treated as 1.0).

Mesquite cannot do likelihood calculations in trees with soft polytomies, or if some taxa have missing data, polymorphisms, uncertain states, or gaps in the character. The calculations also require that the states of a character are contiguous from zero; i.e., the character cannot have only states 0, 1 and 3.

Other programs that reconstruct ancestral states using likelihood are Pagel's Discrete and Swofford's PAUP*.

Making, editing and applying probability models: To use the likelihood calculations, stochastic (probabilistic) models of evolution must be defined. Two models are predefined: a general Mk1 model and a general AsymmMk model. Both of these have their parameters unspecified.

You can also create your own models and specify their parameters by selecting Characters>New Character Model>Markov k-state 1-parameter model (to make an Mk1 model) or Characters>New Character Model>Asymmetrical 2-param. Markov-k model (to make an AsymmMk model). In either case a window will appear in which you can specify the parameters. The Mk1 model allows you to change the rate. Also, you can change the maximum state allowed using a menu item in the Mk1_model menu (e.g., to restrict it to binary characters, choose 1 as the maximum state). The AsymmMk model allows you to change the forward and backward rates, and the maximum state allowed (using a menu item in the AsymmMk_Model menu). You can also choose to express the two parameters in the AsymmMk model as a rate (which controls both forward and backward rates) and a bias (which controls the ratio of forward to backward rates). A bias of greater than 1 means forward changes are more probable; a bias of less than 1 means that backward changes are more probable.

After creating a model, you can edit it by selecting it under Characters>Edit Character Model. You can rename or delete a model by going to the List of Character Models window available under Characters.

Once models are defined they can be applied and used. When setting up a likelihood calculation, if you indicate to use "Stored Probability Model", the calculation will use the selected model for all characters. Alternatively, if the characters used are stored in a matrix (instead of generated temporarily such as by simulations), then each character can be assigned a model in advance of the calculation. This can be done by going to the List of Characters window, selecting the row(s) corresponding to the desired character(s), and then touching on the column heading "Probability Model". A drop-down menu contains a submenu that allows you to select the models to apply. These models will remain assigned to the characters if you save and reopen the file. You can also change the current probability model applied to a character by selecting a module in the Probability Model submenu of the Trace menu. Once models are assigned to the characters, then these are treated as the "Current" models applied to the characters. To indicate that the likelihood calculations use these assigned models, indicate "Current Probability Model" when asked for the source of models.

Optimization Settings: Models used for likelihood calculations may have adjustable settings for the optimization routines used to estimate parameter values. These settings can be changed under Characters>Model Settings; once changed the settings are universal, applying to all calculations with that category of model. They are stored in the preferences directory and used again the next time you start Mesquite. The **Mk1** Model has one setting: The coarseness of the intervals surveyed in optimizing the rate parameter. Wider intervals may result in finding the optimum more quickly when rates are high, but may make less accurate when rates are low. Mesquite by default tries the optimization twice, first with width 1.0 then again with width 10.0, and then chooses the best results. You can request that Mesquite try a single fixed width; suggested values are 1.0 to 20.0. To request Mesquite that use its default strategy, enter a width of 0. The **AsymmMk** model has one setting, concerning the values of parameters used as starting points in the search for optimal parameter values. This setting is used when both parameters of the model are unspecified and need to be estimated. There are three options: (1) The rate is first estimated using the Mk1 model, and then the optimization routine is given that rate plus a bias of 1.0 as starting values. (2) The forward and backward rates of 1.0 and 1.0 are used as starting values, and then 1.0 and 0.1, and then 0.1 and 1.0. Of these three attempts, the parameter values from the attempt yielding the highest likelihood is chosen. (3) Options 1 and 2 are combined, resulting in four attempts to estimate parameters, first using the Mk1 rate then the three alternative forward and backward rates as starting values. Of the four attempts, the parameter values from the attempt yielding the highest likelihood is chosen. The default option is (1). Because the AsymmMk sometimes uses the Mk1 model, changing the setting of the Mk1 model may affect results from the AsymmMk model.

Reporting of results: Likelihood ancestral state reconstructions can be reported in various ways. A first issue is whether only the best estimates are shown at a node or instead the support for each state is shown at a node, regardless of how strong or weak. This is controlled in Trace Character History by the Display Proportional to Weights menu item. If it is selected, then support for each state is shown; otherwise, only the states judged best are shown. The judgment of what are the best states is made according to a decision threshold T, such that if the log likelihoods of two states differ by T or more, the one with lower likelihood (higher negative log likelihood) is rejected. This is set using the Likelihood Decision Threshold menu item.

What states are judged best can be viewed using Trace Character History in several ways: (1) When the cursor is held over a branch and the list of states appears at the bottom of the Trace Legend, the states judged best according to the threshold are marked with an asterisk; (2) In the Text view of the window, the list of reconstructions shows an asterisk by each state judged best at the node; and (3) When Display Proportional to Weights is turned off, only the best states are shaded on the branches. The threshold and best states are also used in Trace Character Over Trees.

Another issue is whether the likelihoods of alternative states are reported as is (Raw Likelihoods) or not. The other two options are to report proportional likelihoods (likelihoods of states are scaled so that they add up to 1, and thus for each state is shown its proportion of total likelihood) or negative log likelihoods. The former are convenient for interpretation and visualization; the latter may be more easily used in statistical tests.

Comparison and Interaction with other Programs

NEXUS-based programs: Mesquite currently saves the models of evolution for likelihood in the private MESQUITECHARMODELS block in NEXUS files. A private block is used because there is as yet no standard for designating such models. Thus, PAUP*, MrBayes and other programs doing likelihood calculations will not be able to access these character models.

Comparison with MacClade: Users familiar with MacClade (macclade.org) will notice some of its features missing from Mesquite, and vice versa. MacClade is restricted to parsimony reconstructions, but has the following features that Mesquite currently lacks. MacClade's Trace Character facility has the ability to fix states at a node (the paintbrush tool) and to show individual MPR's (MPRs mode, formerly Equivocal Cycling). MacClade's Trace All Changes mode and Changes & Stasis chart summarize reconstructed changes in all characters. Parsimony models include Dollo, irreversible and character state trees. Mesquite, on the other hand, includes likelihood reconstructions, reconstructions for continuous characters better integrated with Trace Character, branch-length sensitive calculations and other features such as Trace Over Trees.

Pagel's Discrete and Multistate programs: Pagel's Discrete and Multistate programs also do likelihood reconstructions of ancestral states. Discrete's Fossil Likelihoods with the Global option corresponds to Mesquite's Likelihood Ancestral States. To obtain reconstructions in Multistate equivalent to Mesquite's, define the restricted model equivalent to Mk1 or AsymmMk, using a series of "restrict" commands to set the q parameters equal as appropriate. Use the "test" command to estimate the q rates. Then, restrict the rates to those estimated. Next, use the "fossil" command to fix the node of interest to each of the states in turn, after each using the "test" command to ask Multistate to evaluate the likelihood. These likelihoods are the global likelihoods for the states at the node.

Discrete and Multistate have several features not available currently in Mesquite, including the Local option for parameter estimation, more complete reporting of statistics for the reconstructions, and calculations to test correlation among characters using likelihood ratio tests.

Mesquite can import and export data files for Discrete and Multistate (ppy files). To import, select the file with Mesquite and choose Pagel format in the import dialog box. To export, select File>Export....

Acknowledgments

David Swofford assisted by providing code in C, translated by us to Java, for the optimization routines used in the likelihood reconstruction.

References

- Lewis, P.O. 2001. A likelihood approach to estimating phylogeny from discrete morphological character data. *Systematic Biology* 50:913-925.
- Lutzoni F, M. Pagel & V. Reeb. 2002. Major fungal lineages are derived from lichen symbiotic ancestors. *Nature* 411: 937-940.
- Maddison, D.R. and W.P. Maddison. 2000. MacClade version 4: Analysis of phylogeny and character evolution. Sinauer Associates, Sunderland Massachusetts.
- Pagel, M. 1999. The maximum likelihood approach to reconstructing ancestral character states of discrete characters on phylogenies. *Systematic Biology*. 48: 612-622.
- Pagel, M. 2000. Discrete, version 4.0. A computer program distributed by the author.
- Pagel, M. 2002. Multistate, version 0.6. A computer program distributed by the author.
- Schluter D, T. Price, A.O. Mooers, D. Ludwig. 1997. Likelihood of ancestor states in adaptive radiation. *Evolution*. 51: 1699-1711.
- Swofford, D.L. 2002. PAUP*. Phylogenetic Analysis Using Parsimony (*and Other Methods), Version 4.0. Sinauer Associates, Sunderland, Massachusetts.
-

Processes of Character Evolution

Some characters evolve more quickly than others; some characters depend on others in their evolution. Discovering the nature of these evolutionary processes for a character from an analytical point of view involves determining a model and its parameters.

Contents

- Single characters
 - [Estimating Parameters](#)
- Correlated characters
 - [Visualizations](#)
 - [Pairwise Comparisons](#)
 - [Felsenstein's contrasts](#)
 - [Use with Pagel's Discrete and Multistate](#)

Estimating parameters

Maximum Likelihood estimates of rates and biases can be obtained for categorical characters for two simple models, the [Mk1](#) model and the [AsymmMk](#) model. For more information on these models, see the page on [ancestral state reconstruction](#). Mesquite cannot yet estimate parameters for models of DNA sequence evolution.

Three modules provide calculations to estimate parameters for the Mk1 and AsymmMk models:

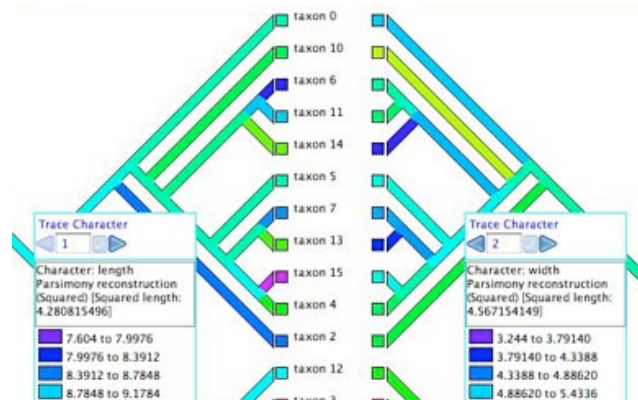
- **Mk1 Estimated Rate**— Estimates the rate of a character's evolution under the simple Mk1 model.
- **Forward/Backward Rates** — Uses maximum likelihood to estimate the rates of forward and backward changes (0 to 1 and 1 to 0 changes respectively), or alternatively the overall rate and the bias in gains versus losses, using the AsymmMk model on a tree for a given character.
- **Asymmetry Likelihood Ratio Test** — Calculates the test statistic for the likelihood ratio test comparing the asymmetrical and one parameter models [$2\ln(L(\text{Asymm.})/L(\text{Mk1}))$], on a tree for a given character.

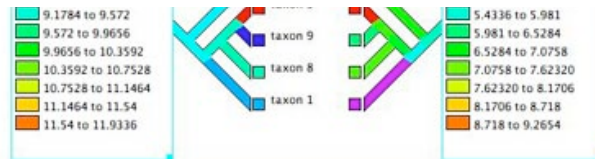
These calculations consider a categorical character and a tree. As such, they can be considered to be values describing a character (and thus are available when analyzing characters, as for instance in a Characters Histogram or Scattergram or a List of Characters Window) or values describing a tree (and thus are available when analyzing trees, as for instance in a Trees Histogram or Scattergram or a List of Trees Window). To access them as values for characters, select "Character value with current tree" or "Character value with tree". To access them as values for trees, select "Tree value using character". Two example files illustrate parameter estimation, Mesquite_Folder/examples/Ancestral_States/15a-estimatingParameters.nex and Ancestral_States/15b-estimatingParameters.nex

Correlations: Visualizations

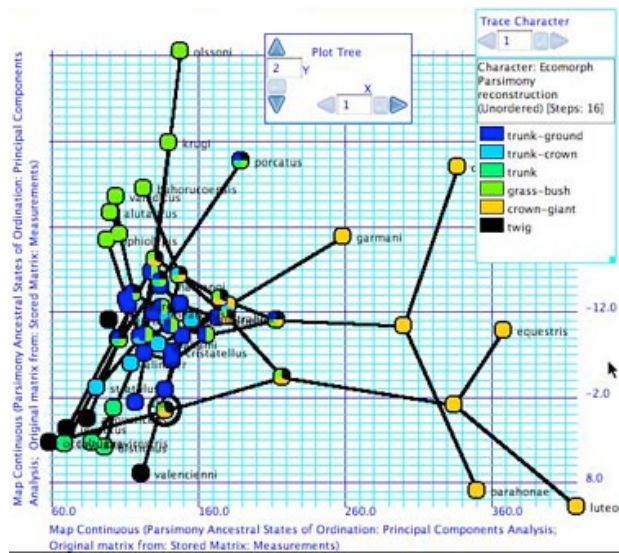
To study correlations or associations among characters, there are both correlation calculators (see [Pairwise comparisons](#), [Felsenstein's contrasts](#), below) and heuristic visualizations. The latter include:

- **Mirror Tree Window** — When a Tree Window is open, you can request an alternative view of the same tree by selecting [Tree>Mirror Tree Window](#). This shows the same tree as in the tree window, shown in duplicate tips-to-tips. The purpose of this is to allow you to display two different visualizations (one at left, one at right) and compare them. Character correlations can be explored by tracing evolution of two characters, as shown here.
Example files: Basic_Examples/tree_viewing/08-mirrorTree.nex; Ancestral_States/15-Mk1AsymmCompare.nex; Pairwise_Comparison/01-pairwise.nex.

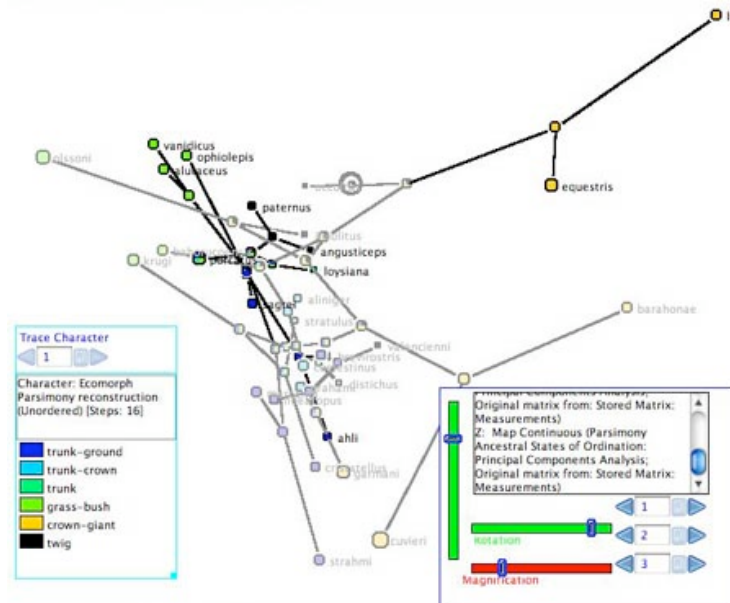




- **Plot Tree 2D** – Plots the tree in a 2-dimensional space, available as a tree drawing form in the Drawing>Tree Form submenu. If the axes represent the state of the taxa in two continuous characters, then this allows one to map the tree into the character space, which may suggest patterns or correlations. The internal nodes of the tree can be placed at the reconstructed ancestral states. An example is shown here.
Example file: Multivariate_Continuous/07-anoles.nex



- **Plot Tree 3D** – Plots the tree in a 3-dimensional space, available as a tree drawing form in the Drawing>Tree Form submenu. This is part of the Rhetenor package. As with Plot Tree 2d, this allows one to map the tree into the character space. The tree can be rotated in space using the Rotation sliders in the legend. An example is shown here.
Example file: Multivariate_Continuous/08-anoles.nex



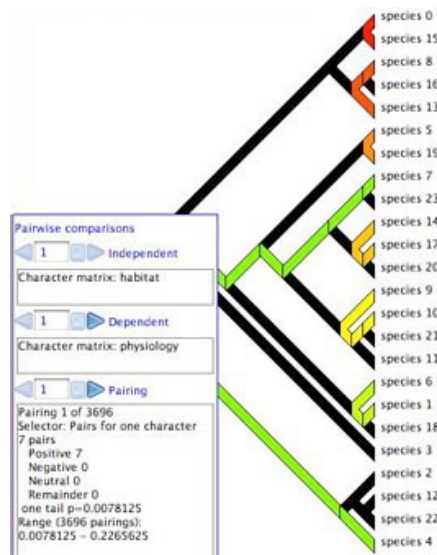
- **Taxa Scattergram** – Select Analysis>New Scattergram For>Taxa to obtain a bivariate plot for taxa. You will be asked whether to use the same or different calculations for the two axes. By "Different" is meant two entirely different calculations, such as the percentage of missing data in the taxon on one axis, and the state of a continuous variable on the other. Choose "Same" and then, if asked, indicate you want "Continuous state of taxon". You will therefore be plotting the taxa according to their states in one character versus another. If [PDAP](#) is installed, you will be able to do linear regression by selecting Scattergram>Analysis>Other Choices..., and choosing one of the Scattergram Diagnostics. Note: any correlation seen is aphylogenetic. Phylogenetic correlations can be studied by

using the [Felsenstein's contrasts](#) calculations in PDAP.
Example files: Multivariate_Continuous/01-wingsPlot.nex and subsequent

Correlations: Pairwise comparisons

Character correlations can be tested using pairwise comparisons as described by Read & Nee (1995) and W. Maddison (2000). This is available under the Analysis menu of Tree Windows. The module chooses pairs of taxa, and indicates how the pairs compare in two characters: does the member of the pair with the higher value (say, state 1) in one character have higher or lower value in a second character? A summary over all pairs is given in the legend, as shown below. There are three options for choosing pairs:

- Most pairs – choose pairs to maximize number of pairs, regardless of the states in the characters
- Pairs for one character – choose pairs of taxa that differ in the state of the first character (independent variable)
- Pairs for two characters – choose pairs of taxa that differ in the state of both characters

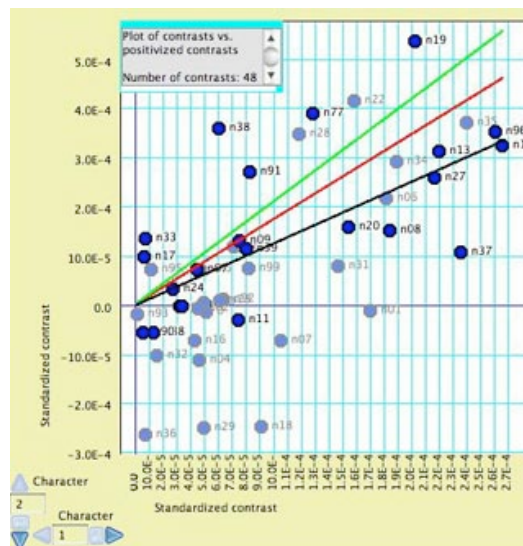


The graphical display shows the current pairing chosen; you can scroll through all pairings using the legend.

Example files: Pairwise_Comparison/01-pairwise.nex and subsequent

Felsenstein's Independent Contrasts

Correlations among continuous valued characters can be studied using the separately-available [PDAP](#) package (Midford et al., 2003), which (among other things) calculates Felsenstein's (1985) independent contrasts and displays them in a scatterplot:



The points in the plot are nodes in the tree, with the X and Y axes representing the independent contrast across the node in each of the two characters. Regression lines, confidence intervals and other statistics can be calculated by PDAP. When only some nodes in the tree are selected, they are highlighted in the

plot as shown above.

The PDAP documentation or example files should be consulted for more details.

Use with Pagel's Discrete and Multistate

Mesquite does not yet perform Pagel's (1994) test for correlation among categorical characters, but it can import and export files for use by the Discrete (Pagel, 2000) and Multistate (Pagel, 2002) programs. For import, attempt to read the Pagel format file in Mesquite, and choose the file format from the import dialog box. For export, select the Export... menu item from the File menu.

References

- Felsenstein, J. 1985. Phylogenies and the comparative method. *American Naturalist*, 125:1-15.
- Maddison, W.P. 2000. Testing character correlation using pairwise comparisons on a phylogeny. *J. Theoretical Biology*. 202: 195-204.
- Midford, P. E., T. Garland Jr. & W. Maddison. 2002. PDAP:PDTree package for Mesquite, version 1.00.
- Pagel, M. 1994. Detecting correlated evolution on phylogenies: a general method for the comparative analysis of discrete characters. *Proc. R. Soc. London B* 255: 37-45.
- Pagel, M. 2000. Discrete, version 4.0. A computer program distributed by the author.
- Pagel, M. 2002. Multistate, version 0.6. A computer program distributed by the author.
- Read, A. F. and S. Nee. 1995. Inference from binary comparative data. *J. Theoretical Biology* 173:99-108
-

Statistical inference using simulations or randomizations

Mesquite has various tools to simulate and randomize characters and trees, including simulations of DNA sequence evolution, tree reshuffling, coalescent gene tree simulations, and others. With these tools, you can build your own statistical inference methods tailored to your question.

For instance, you can test an hypothesis about phylogenetic structure using parametric bootstrapping, simulating many character matrices on a proposed tree, for each attempting to reconstruct the tree and assessing the reconstruction. Or, you can simulate gene trees within a population to test a population genetics or phylogeographic hypothesis. Or, you can study the effect of branch length uncertainty on calculation of Felsenstein's contrasts by adding random noise to branch lengths on a given tree, repeating many times to see the variation in calculated correlation coefficient of contrasts in two characters. Many other analyses are possible. We give an overview of the features and some examples in the following.

- Simulating and randomizing characters
 - [Overview](#)
 - [Simulating DNA sequence evolution, with examples of hypothesis testing](#)
 - [Simulating coalescence and sequence evolution within populations](#)
 - [Simulating and randomizing trees](#), including species trees and gene trees
-

Character simulations and randomizations

Mesquite can simulate and randomize characters to build statistical tests. On this page we give an overview of these features. A more in-depth account of simulation of DNA sequence evolution is given [separately](#).

Contents

- [Using results of simulations & randomizations](#)
- [Simulations of character evolution](#)
- [Viewing results of simulations](#)
- [Randomizing characters](#)

Using results of simulations & randomizations

The simulated or randomized characters can be used or stored in several ways:

- The characters can be stored into matrices in the current file by choosing options in the **Make New Matrix from** submenu of the Characters menu. For instance, if you choose Simulated Matrices on Current Tree, the matrix simulated will be stored in the file.
- The characters may be used directly, at that moment, in calculations. For example, if you make a Histogram for Characters, and choose Simulated Characters as your source of characters, the characters will be simulated and used in the chart without being stored in the file.
- A series of many data files can be saved, each one with a different replicate of the simulated or randomized data matrix. This is available through the **Save Multiple Matrices** submenu of the Character menu
- A series of many data files can be saved in combination with scripting files to instruct programs such as Swofford's PAUP to run the files. This can be done using the **Batch Architect**, a description of which is in the page on [DNA simulations](#) and some of the [Studies](#).

To replicate the results of a simulation or randomization, you can use the **Set Seed** menu item to set the random number seed used. If you are using the same conditions, including the same seed, the simulations and randomizations should be reproducible.

Simulations of character evolution

Stochastic models can be used to simulate character evolution along the branches of a phylogenetic tree by selecting Simulated Characters (to generate characters one at a time) or Simulated Matrices on Current Tree (to generate whole matrices, on a current tree in a Tree Window), or Simulated Matrices on Trees (to generate whole matrices, each one on a different tree from a source of trees). These options are available whenever characters or matrices might be called for, for instance when making a chart of characters or matrices.

The following are the character types and models that can be simulated:

- **Evolve Categorical characters.** The following models are also discussed in the section on [likelihood reconstructions](#).
 - **Mk1 model** — Single parameter model analogous to Jukes-Cantor. Rates of change equal for all types of state-to-state changes.
 - **AsymmMk model** — Two parameter asymmetrical model with differing rates of forward and backward changes. Forward changes are those in which state number increases (e.g., state 0 to state 1); backward changes are those in which state decreases (e.g., state 1 to state 0). One can specify the forward and backward rates directly, or alternatively, one can specify an overall rate of change in combination with a bias of forward versus backward. This model will generally be appropriate only for binary characters.
- **Evolve DNA characters**
 - See page on [DNA simulations](#) for details.
- **Evolve Continuous characters**
 - **Brownian motion model** — Model with a single parameter, the rate of change.

To use these simulations, the appropriate character model must be defined in advance, with all of its parameters specified. Two models come built-in: a Jukes-Cantor model for DNA simulations, and a Brownian motion model for continuous variable simulations. If you want any other models, created them using **New Character Model** in the Characters menu.

Viewing results of simulations

Simulated characters can be used in many calculations, but if you want to visualize directly the results of a simulations you can use the **Trace Character History** feature available in the Analysis menu of the Tree Window. By default Trace Character History shows a reconstruction of ancestral states. Thus, if the character is simulated, the states at nodes shown would not be the "true" ancestral states that occurred

during the simulation, but rather states inferred from the states given to the terminal taxa by the simulation. However, once Trace Character History is active, its Trace menu has a **Character History Source** menu item. Choose **Simulate Ancestral States** and specify the simulation. The states indicated at the nodes will then be the true ancestral states in the simulation. You can set the Seed to make the simulation equivalent to simulations done in other contexts.

Randomizing characters

Existing characters can be randomized as follows:

- Reshuffle Character — Supplies replicate reshufflings of a single chosen character. In each reshuffling, the character states are randomly scrambled among taxa, keeping the frequencies of different character states fixed.
 - Reshuffle Matrix — Supplies matrices, each of which is a reshuffling of an existing matrix. The first character of the matrix is a reshuffling of the first character of the original matrix; the second character is a reshuffling of the second original character; and so on.
 - Bootstrap resample — Supplies matrices, each of which is a bootstrap resampled version of an existing matrix. Characters are sampled randomly from the original matrix and moved into the resampled matrix until it contains as many characters as were in the original matrix. Some of the original characters may by chance be sampled more than once; some may be not sampled at all.
 - Rarefy characters — Supplies matrices, each of which is derived from an existing matrix by randomly deleting entire characters.
 - Sprinkle missing — Supplies matrices, each of which is derived from an existing matrix by randomly assigning "missing" (?) or unassigned) to cells of the matrix with a particular probability.
 - Add noise (for continuous matrices only) — Available in the Character Matrix Editor under Alter/Transform, this adds noise to the states of all or selected cells of the matrix.
 - Random Fill — Available in the Alter/Transform of the Matrix menu of the Character Matrix Editor, it can be used to fill all or selected cells of the matrix with randomly chosen states.
-

Simulating DNA sequence evolution, with examples of hypothesis testing

This section provides a brief introduction to some of the analyses you can do by simulating DNA sequence evolution, including:

- [Simulating DNA sequence evolution on a tree](#)
- [Building more complex models](#)
- [Batch Architect: Simulating, exporting, and analyzing multiple matrices](#)
- [Examples of tests of phylogenetic hypotheses](#)

Example files for this chapter are in the "Character_Simulations" folder within the "examples" folder within Mesquite_Folder.

Simulating DNA sequence evolution on a tree

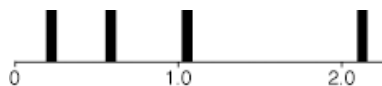
First, let's try evolving DNA sequences up the branches of a model tree, thereby creating an entire simulated matrix. Open the example file "01-modelTree.nex" and you will be presented with a tree. For the moment it doesn't matter where this tree came from; the important thing to note is that it has branch lengths inferred from a matrix of 18S rDNA; this matrix is also included in the file.

In order to simulate DNA sequences, you need:

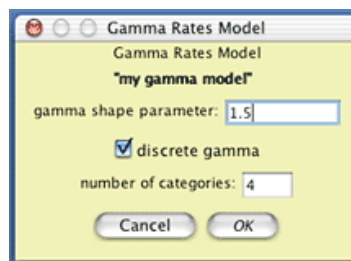
- A tree, with branch lengths specified. A tree with branch lengths is included in this example file.
- A specification of how the DNA sequences evolve. This model could have many elements. One common form of model specifies the relative frequencies of A, C, G, and T; the relative rates of change of the different characters; and the relative rates of change between A and C, A and G, and so on.

Creating a model of character evolution

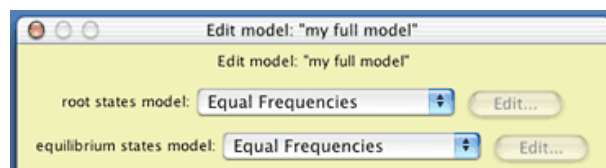
For sake of simplicity, let's presume that A, C, G, and T occur at equal frequencies, and that all pairs of nucleotides have equal rates of change between them. To make it interesting, though, let's presume that some characters evolve more quickly than others, with the distribution of rates of change following a discrete gamma distribution with shape parameter 1.5. If one presumes (as we will) that there were four categories of rates, then this model presumes that one quarter of the characters have rate 0.225, one quarter 0.589, one quarter 1.05, and the rest 2.136, as shown in the following figure:

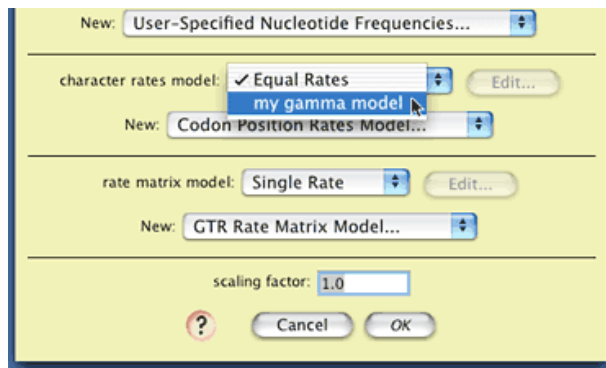


To create this model of character rate variation in Mesquite, choose (Tree) Characters>New Character Submodel>Gamma Rate Variation Model... You will be presented with a dialog box in which you can choose the name of the model you will create; you might call it "my gamma model". You will then be presented with a dialog box allowing you to specify the nature of the model; set the shape parameter to 1.5:



Now we need to create a composite model of character evolution that specifies both this gamma rate variation model, the state frequency model, and the state-to-state rate model. To do this, choose (Tree) Characters>New Character Model>Composite DNA Simulation Model..., name the model (perhaps call it "my full model"), and you will be presented with a dialog box in which you can choose the various submodels:





The only element of this model we will change is the character rates model; choose "my gamma model" from the drop-down menu to select the gamma distribution model you created as the model of site-to-site rate variation. The model "my full model" is now ready to use.

Note that the dialog box in which you edited "my full model" has button with a ? on it (?). When a button like this is present in a Mesquite window, touching on it will display a small note with information about how to use the window's feature.

Simulating evolution

Now choose (Tree) Characters>Make New Matrix from>Simulated Matrices on Current Tree. This will ask Mesquite to simulate evolution of characters up the current tree in the tree window to produce a new matrix. You will be asked what sort of character simulator to use. As we wish to evolve a DNA sequence up the tree's branches, choose "Evolve DNA characters" from the list. (There might be other options presented, such as Evolve Continuous Characters, but we don't wish to try that now.) You will then need to chose the model of DNA sequence evolution; choose the composite one you created earlier called "full model". When it queries for the number of character, choose a large number, such as 2000. When it asks for the name of the matrix to be created, call it "simulated" (although you could choose another name, if you wish). Mesquite will then simulate evolution, and a matrix will be produced, such as this one:

Taxon \ Character	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43
1 Omus	C	T	G	T	A	T	G	A	G	T	A	A	C	G	C	G
2 Oxycheila	C	T	G	T	A	T	G	A	G	T	A	C	C	G	C	G
3 Clinidium	C	T	G	T	A	A	G	C	G	T	A	T	C	C	G	
4 Omoglymmius	C	T	G	T	A	A	G	C	G	T	A	A	C	T	C	G
5 Trachypachus	C	T	G	T	A	A	G	A	G	T	A	A	C	G	C	G
6 Systolosoma	C	T	G	T	A	A	G	A	G	T	A	A	C	G	C	G
7 Metrius	C	T	G	T	A	A	G	A	A	T	C	A	C	G	C	G
8 Pachyteles	C	T	G	T	A	A	G	A	A	T	A	A	C	G	C	G
9 Scaphinotus	C	T	G	T	A	T	G	A	G	T	A	A	C	G	C	G
10 Nebria	C	T	A	T	A	T	G	A	G	T	A	A	C	G	C	G
11 Loricera	C	T	G	T	A	A	G	A	G	T	A	A	C	G	C	G

Examining the results of the simulation

Let's see whether the rate variation between characters suggested by this simulated matrix matches a gamma distribution with shape parameter 1.5, as used in the model. First, save the matrix to a file so that you can read it into PAUP*. Choose (Character Matrix) Characters>Save Copy of Matrix>simulated to save a copy of your newly created matrix; perhaps name the file "Simulated Matrix".

In PAUP* 4, execute the file Simulated Matrix, and then execute the following commands:

```
nj;
lset nst=1 basefreq=equal rates=gamma shape=estimate;
lscore 1;
```

(If you don't know how to give PAUP* commands, please read your PAUP* documentation. There are equivalent ways to give these commands in the MacOS versions using dialog boxes, but it is much easier for us to describe what to do with these text commands.)

The first command ("nj;") will find a neighbor-joining tree. While this might not be the best way to get a tree, but it will be fast, and give a tree that is good enough for our purposes. The second command ("lset...") sets the model used for likelihood calculations to be the model used by Mesquite in the simulations, except that the shape parameter used is not specified, but is instead estimated from the

data. The third command ("lscore 1;") tells PAUP* to calculate the likelihood of the tree, in the course of which it will also estimate the shape parameter of the gamma distribution using likelihood. You might end up with PAUP* reporting something like this:

```
Tree 1
-----
-ln L 19742.05025
Shape 1.583552
```

The shape parameter should be around 1.5.

Building more complex models

More complex composite models of DNA evolution can be built by choosing among the following:

Models of root state frequencies:

- Equal Frequencies: equal frequencies of states
- Empirical Frequencies: frequencies of states the same as that found in an existing character matrix
- User-specified Nucleotide Frequencies: frequencies of nucleotides specified by the user. You can create a model of this sort by choosing ([Character Matrix](#)) [Characters>New Character Submodel>User-specified Nucleotide Frequencies...](#) and then entering the appropriate values in the dialog box.

Models of equilibrium state frequencies on other branches:

- Equal Frequencies: equal frequencies of states
- Empirical Frequencies: frequencies of states the same as that found in an existing character matrix
- User-specified Nucleotide Frequencies: frequencies of nucleotides specified by the user. You can create a model of this sort by choosing ([Character Matrix](#)) [Characters>New Character Submodel>User-specified Nucleotide Frequencies...](#) and then entering the appropriate values in the dialog box.

The above two submodels corresponds in PAUP* to the specifying the base frequencies in the likelihood settings; note that PAUP* does not allow separate specification of the model of frequencies and the root and at other branches.

Models of rate variation among characters:

- Equal Rates: all characters evolve at the same rate.
- Codon Position Rates Model: a model specifying the relative rates of the different codon positions. If you wish to have the codon positions match those in an existing matrix, then codon positions need to be specified for a DNA matrix. To do this, choose ([Character Matrix](#)) [Characters>List of Characters](#) to see the current codon positions. Select the characters in this list window, and then touch on the title of the column "Codon Position". A drop-down menu will appear in which you can choose to set the codon positions to a sequence like 123123123... as appropriate.
- Gamma Rates Model: a model specifying that rates of characters evolve according to a gamma distribution.
- Gamma Invar Rates Model: a model specifying that a proportion of the characters are invariant, and the remainder follow a gamma distribution.
- Proportion Invariant Model: a model specifying that a proportion of the characters are invariant, and the remainder evolve at one rate.

The first model ("Equal Rates") is built in; you can create a model of the other kinds by choosing the appropriate items from the ([Character Matrix](#)) [Characters>New Character Submodel](#) menu and then entering the values in the dialog box.

Model of rate matrices:

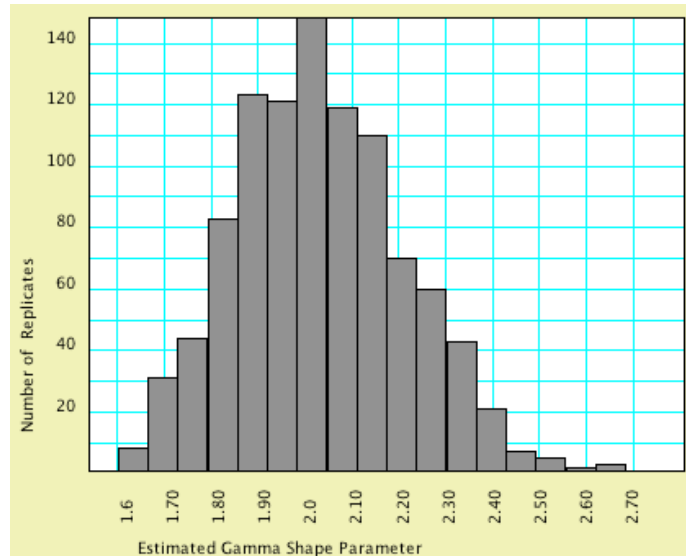
- Single Rate: all changes between nucleotides occur at the same rate.
- Ti/Tv Rate Matrix Model: the two-parameter rate matrix model in which transversions occur at a different rate than transitions.
- GTR Rate Matrix Model: the General Time Reversible, six-parameter rate matrix model in which you can specify rates of each type of nucleotide change.

The first model ("Single Rate") is built in; you can create a model of the other kinds by choosing the appropriate items from the ([Character Matrix](#)) [Characters>New Character Submodel](#) menu and then entering the values in the dialog box.

Batch Architect: Simulating, exporting, and analyzing multiple matrices

Mesquite's **Batch Architect** package can be used to create multiple simulated matrices, export them to files, and create one or more files that provide instructions to programs to analyze the files produced. These instruction files are called "batch files".

One could, for example, examine the quality of Mesquite's simulation algorithms by simulating 100 matrices using a gamma model of character rate variation. In addition to creating 100 files with matrices, Mesquite would create a batch file containing instructions for PAUP* to estimate the gamma shape parameter for each of the matrices and save the resulting values to a score file. Mesquite would also create an file that would provide instructions to Mesquite to allow it to read PAUP*'s score file and display the resulting distribution of estimated gamma shape parameters in a chart. As a concrete example, simulation of 1000 matrices of 2000 characters each under a model with a gamma shape parameter of 2.0, estimating of the shape parameter by PAUP* for each of these matrices, and processing of the results by Mesquite yielded the following histogram:



In this particular analysis, most matrices were estimated to have a gamma shape parameter close to that used in the model that generated the data; the average estimated gamma shape parameter over the 1000 replicates was 2.017.

Conducting multiple simulations

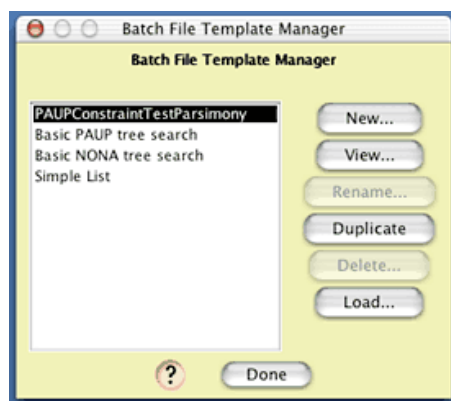
To conduct an analysis similar to this one, open up the example file "**02-gammaTest.nex**". A complete model of DNA sequence evolution has already been added to that file, under the name "gamma2model". It specifies, among other components, character rate variation following a gamma distribution with shape parameter 2.0. First, you will need to ask Mesquite to evolve several matrices up the branches of the phylogeny. Choose (Tree Window) Analysis > Matrices & Batch Files > Export Matrices & Batch Files.... You will be presented with a choice of the source of matrices; choose "Simulated Matrices on Current Tree", and press OK; for the character simulator, choose "Evolve DNA characters". When asked to choose the model, choose "gamma2model".

You will now be presented with the main dialog box that you will need to master to use Mesquite's simulation tools. It looks approximately like this:



The top part of the dialog box allows one to choose the base part of the name of the simulated matrices that will be produced, as well as to specify how many matrices will be produced (the "number of replicates"). Let's call have the file names begin with "gammaTest", so enter that for the base name. In the interests of time, you can leave the number of replicates be only 10 (a more thorough analysis would involve many more replicates).

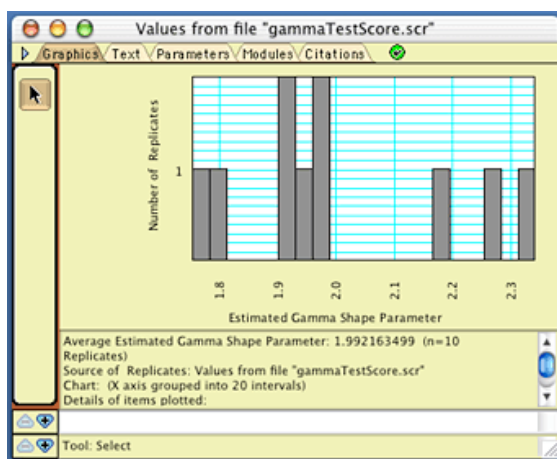
In addition to creating the matrices, Mesquite will also produce one or two files accompanying those matrices that specify commands to be used by various software (e.g. PAUP, NONA, even Mesquite) to analyze those matrices. The manner in which these accessory files or "batch files" are written is controlled by the lower portion of the dialog box. The content of the batch files is specified by a "batch file template". There are some templates supplied by Mesquite; you can also build your own. The template we need to use for this example is not built into Mesquite; you will need to load it in from a file. To do this, go into the Templates Manager, by pressing the Edit Templates button. In the dialog that is presented, you can see the build-in templates:



To load in the template we need to use, press the Load button, and open the "GammaTest.template" file that is in the "templates" folder within the Simulations examples folder. The template Gamma Model Test will now be added to the templates list. (You can see the nature of the template by selecting it in this list and pressing "Edit", but you needn't do this now.)

Once the template is loaded, press Done to return to the Export Matrices & Batch Files dialog box. Make sure "Gamma Model Test" is selected, and press "OK". You will now be asked to choose a folder to store the created files. As multiple files will be created, you may wish to create a new, empty folder to house the files. Once the location is chosen, you will be asked (after perhaps dismissing a notice or two) for the number of characters to be simulated within each matrix. Choose a fairly high number, such as 2000. Mesquite will now proceed to create through simulation all 10 files, plus two batch files.

One of these batch files will be a file containing PAUP commands that will instruct PAUP* to read in each matrix file, quickly build a preliminary tree (using neighbor-joining, which should be good enough for our purposes), and estimate the gamma shape parameter using likelihood. The resulting value will be saved to a PAUP "scorefile". Open up PAUP* 4 (make sure you have the latest version of PAUP*), and ask it to execute the file "gammaCommands.nex". After PAUP has finished its analysis, go back to Mesquite, and choose (Tree Window) Analysis > Batch Architect> Show Results via Instruction File.... You will be asked to first choose an "instruction file", which is a text file telling Mesquite how to interpret another file. Choose the file "MesquiteInstructions", which is the second batch file created by the Gamma Model Test template. After Mesquite processes the instruction file, it will ask you to find the results file, which in this case is the score file created by PAUP*. It should be called "gammaTestScore.scr" (if the base name of the matrices was "gammaTest"); open it when asked for the results file. You will then be presented with a histogram of the estimated gamma shape parameter values, that will look approximately like this:



Of course, your exact values should differ from these, but should cluster around 2.0. If you repeated this, but with many more replicates, then you should see a bell-shaped curve, as shown above.

Testing simulations of more complex models

A test of accuracy of Mesquite's more complex models can also be done. For example, there are seven parameters in model that includes a General Time Reversible model of DNA evolution and assuming a proportion of the characters are invariant with the remainder following a gamma distribution. In 1000 replicates of 2000 characters evolved up the branches of a 49-taxon tree, the average value of these parameters as estimated by PAUP* is close (within 1.3 %) of the value used in model used in Mesquite, as shown by the results of one particular analysis:

Parameter	Value in model	Estimated value	Difference
Rate(A<->C)	1.8700052	1.87771	0.41%
Rate(A<->G)	4.2537253	4.26319	0.22%
Rate(A<->T)	2.5286454	2.53006	0.06%
Rate(C<->G)	0.626305	0.63152	0.83%
Rate(C<->T)	8.735118	8.77468	0.45%
proportion invariant	0.506903	0.50903	0.42%
gamma shape	0.438522	0.44422	1.30%

Examples of tests of phylogenetic hypotheses

You can use Mesquite's Batch Architect and Genesis packages to build your own statistical tests of various phylogenetic hypotheses. Some example tests are presented in the following pages:

- [Testing monophyly of a group of beetles](#)
- [Are strepsipterans related to flies? Exploring long branch attraction](#)

Other analyses that Mesquite can do

The examples given about used the Genesis package in Mesquite to simulate the evolution of characters under a specified model of DNA evolution, and up the branches of the current tree on the screen. Many more sorts of analyses can be done, using different sources of characters other than simulated DNA data, different sources of trees other than the current tree on the screen, and different calculations by other programs.

Some of the different sources of character matrices include:

- Simulators that evolve data other than DNA data, including continuously valued data.
- Random modifications of existing matrices, including by non-parametric resampling (as used in classic bootstrap methods), random reshuffling of data, and jackknifing.

Some of the different sources of trees include:

- Simulations using markovian models of speciation
- Coalescence simulations of gene trees
- Modifications of an existing tree by random modification of branch lengths
- Modifications of an existing tree by randomly pruning a fraction of the taxa or by randomly adding taxa
- Randomly sampled trees among all possible trees
- Trees generated by randomly reshuffling taxa at their tips

References

Shull, V., A.P. Vogler, M.D. Baker, D.R. Maddison, and P.M. Hammond. 2001. Sequence alignment of 18S ribosomal RNA and the basal relationships of adephagan beetles: Evidence for monophyly of aquatic families and the placement of Trachypachidae. *Systematic Biology*, 50:945-969.

Tree simulations and randomizations

Various packages in Mesquite can simulate or randomize trees, allowing you to build statistical tests based on null distributions or trees generated under some hypothesis.

Contents

- [Using results of simulations & randomizations](#)
- [Evolutionary Simulations of Trees](#)
 - [Species Trees](#)
 - [Gene Trees](#)
- [Random Trees](#)
- [Randomized Trees](#)

Using results of simulations & randomizations

The simulated or randomized trees can be used or stored in several ways:

- The trees can be stored into tree blocks in the current file by choosing options in the **Make New Trees Block from** submenu of the Taxa&Trees menu.
- The trees may be used directly, at that moment, in calculations. For example, if you make a Histogram for trees, and choose Simulated Trees as your source of trees, the trees will be simulated and used in the chart without being stored in the file.
- A series of many files can be saved, each one with a different replicate of a block of simulated or randomized trees. This is available through the **Save Copies of Tree Blocks** submenu of the Taxa&Trees menu menu

To replicate the results of a simulation or randomization, you can use the **Set Seed** menu item to set the random number seed used. If you are using the same conditions, including the same seed, the simulations and randomizations should be reproducible.

Evolutionary Simulations of Trees

Species Trees

- **Uniform speciation (Yule)** (trees package) – Generates tree by simple uniform probability speciation (a Yule process). The chance of speciation is equal for all tips. Options: total time depth of tree.
- **Uniform speciation with sampling** (TreeFarm package) – Generates a tree by Yule process, as above, but with to a total number of species greater than in the taxa block. Extra species are then randomly sampled out, to leave the tree with the appropriate number of species. Barraclough & Nee (2001) discuss how this sampling alters the branch length distribution of the tree.

Gene Trees

- **Coalescent Trees** (coalesce package) – Generates tree by coalescence within a single panmictic population. Options: Effective population size.
- **Coalescence Contained within Current Tree** (coalesce package) – Generates tree by a simple coalescence model of a neutral gene with constant population size, within a current species tree from a Tree window or other tree context. Branch lengths are assigned according to generation of coalescence. The species tree used is a current tree found in a Tree Window or other tree context.

Random Trees

- **Equiprobable Trees** (trees package) – Generates trees randomly so that each possible labelled tree topology is equally likely.
- **Randomly Resolve Polytomies** (TreeFarm package) – Randomly resolves polytomies in tree. All possible resolutions are equiprobable. Thus, if the tree is a polytomous bush, the resulting resolved trees will be distributed equivalently to that from the Equiprobable Trees module.

Randomized Trees

- **Reshuffle Terminal Taxa** (TreeFarm package) – Shuffles (permutes) the taxa among the terminal nodes.
- **Random Branch Moves** (TreeFarm packages) – Performs a specified number of randomly -chosen branch moves.
- **Add Noise to Branch Lengths** (TreeFarm package) – Adds noise to branch lengths of tree. Noise is Normally distributed with variance specified by user. Optionally, this variance is proportional to current branch length. Negative branch lengths are not allowed, and are changed to

zero. Options: variance of noise.

- **Rarefy Tree** (TreeFarm package) – Rarefies tree by randomly deleting taxa. Options: how many excluded.
- **Augment Tree Randomly** (TreeFarm package) – Augments tree by random placement of excluded taxa. Options: whether branch lengths are to be considered; whether addition is only to original branches

References

Barracclough, T.G. & S. Nee 2001. Phylogenetics and speciation. *TRENDS in Ecology and Evolution*. 16:391-399.

Population Genetics

The calculations in Mesquite's standard packages that concern population genetics include coalescence simulations and calculations involving gene trees. As yet, there are few of the traditional population genetics calculations (e.g., no F_{st}). Some of the relevant features and calculations are:

- simulations of gene trees by coalescence, either within a single population or in a diverging population or species tree
- simulations of sequence evolution, which can be used to evolve haplotypes on a gene tree
- calculations of fit of a gene tree to a population tree or population subdivision (Slatkin & Maddison's s , Maddison's deep coalescences)
- searching for population trees that optimize fit of gene trees
- cluster analysis of populations using similarities of contained gene sequences
- charts, scripting and production of batch analyses to yield statistical tests

In these analyses individual samples or haplotypes will generally be represented by taxa. Thus, each gene sequence will be a taxon, and the sequence itself will be a row in a DNA matrix.

Many of the features and calculations described below are illustrated in the example data files in the Mesquite_Folder/examples/Coalescence/ folder. The majority of these features are provided by the Coalescence package (mesquite.coalesce) and the taxa association package (mesquite.assoc); some are provided by the distance analysis package (mesquite.distance).

Contents

- [Importing and Exporting Data](#)
- [Single populations: Coalescence simulations](#)
 - [Example: Chart of coalescence depths](#)
- [Multiple populations](#)
 - [Establishing the association between genes and populations](#)
 - [Step-by-step method](#)
 - [Automated method](#)
 - [Editing already-created taxa associations](#)
 - [Simulating coalescence within a population tree](#)
 - [Reconstructing gene history within population history](#)
 - [Visualizing gene history in population history](#)
 - [Measuring fit between genes and populations](#)
 - [s](#)
 - [Deep coalescences](#)
 - [Example: Effect of population divergence time on s](#)
 - [Inferring the population or species tree](#)
 - [Tree search](#)
 - [Cluster analysis](#)
- [Simulating sampled gene sequences](#)
 - [Generating a single matrix of sequences](#)
 - [Generating a series of matrices](#)
 - [Example: Multiple simulations of sequence samples](#)
- [References](#)

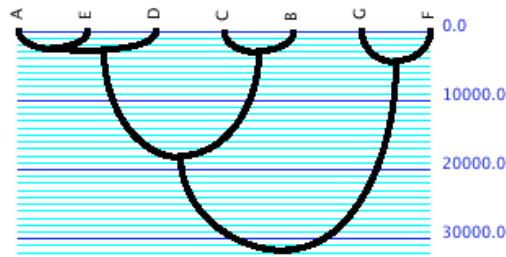
Importing and Exporting data

Mesquite can import and export gene sequences in text files in the following formats: Simple text table, NBRF/PIF, PHYLIP, .ss (NONA, Hennig86, WinClada) as well as NEXUS files.

Single population: Coalescence simulations

Gene trees within a single population can be simulated under the assumption of neutrality, panmixia and constant population size. A sample of such simulated gene trees can help you generate null expectations in tests, for instance of population subdivision. Mesquite simulates these gene trees by a coalescent process, beginning with the set of defined genes (taxa) and coalescing back in time until a single common ancestor is reached. Simulated coalescent trees can be viewed or used in other contexts where a source of trees is used — in the Tree Window or in Trees charts, for instance.

To view simulated gene trees, first prepare a data file with taxa representing the sampled gene copies. Select New Tree Window from the Taxa&Trees menu, and indicate you want Simulated Trees as your tree source. Choose Coalescent Trees as your tree simulator. You will be asked to indicate an effective population size. The simulated tree that appears will probably not be shown with branch lengths indicated; for optimal viewing we suggest indicating you want "Branches Proportional to Lengths" in the Drawing menu, and use the Tree Form of Curvogram. You may also want to choose a narrower line width. The tree might look something like this:



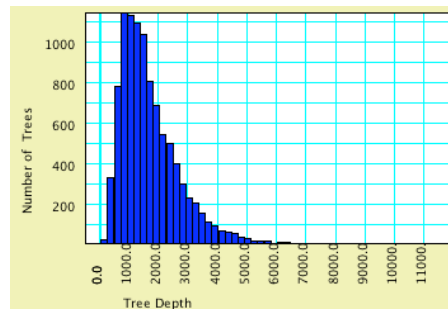
The blue numbers represent time in generations. To examine other simulated gene trees, scroll from one to another using the blue arrows in the upper left corner of the Tree Window. You will notice that depth of the trees vary, depending on when the last coalescence happened during the simulation. In order to fit the tree into the window nicely, the scale of generations changes. You may find it interesting to prevent this rescaling, so that all trees are shown to the same scale. You can do that using the "Fixed Scaling" menu item in the Drawing menu.

The parameters of the simulation may be changed using the Set Seed (Tree simulation) menu item, and the items in the Coalescence Simulations submenu. To change effective population size, select "Set Ne". These simulations treat the organisms as **haploid**. For reasonably large population sizes, an exponential approximation can be used in the simulations to avoid having to model all genes in the population explicitly. This exponential approximation is the default; you can turn it off using the "Exponential approximation" menu item.

Simulated gene trees can be generated and saved in a trees block in your data file. To do this, select Taxa&Trees>Make New Trees Block From>Simulated Trees>Coalescent Trees. Simulated gene trees can also be used in charts, as in the following example.

Example: Chart of coalescence depths

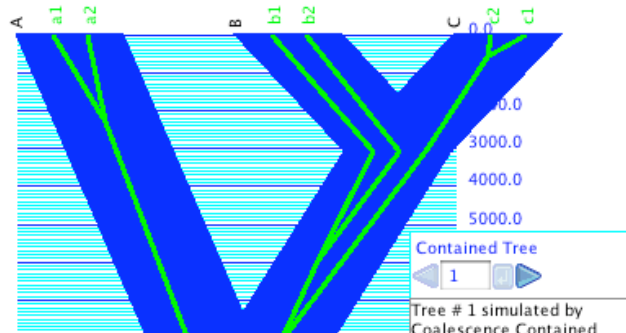
Simulated gene trees can be examined in charts by using Simulated Trees as your tree source. For instance, the example file 03-coalescenceDepth.nex shows a chart of time to final coalescence in a sample of 100 gene trees. This was made by asking for the Histogram for trees, using Tree Depth as the value to calculate (this is a secondary choice), and Simulated Trees, Coalescent Trees as the source of trees. Here is the same chart modified to ask for a sample of 10,000 gene trees.

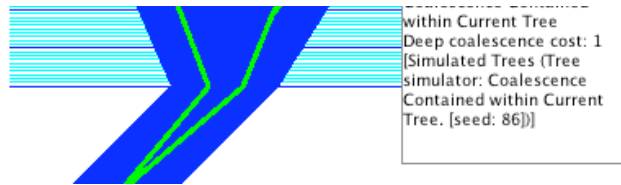


The effective population size is 1,000. Although the average time to final coalescence is about 1760 generations, the curve (as expected) has a long tail. By touching on the Text tab of the window you can see that there was one simulated tree with depth >11,100 generations.

Multiple populations

Gene sequences from multiple populations can be analyzed and modeled in Mesquite. For instance the following shows a simple example of 6 genes within 3 extant populations, in which a coalescence simulation within a diverging population history (blue) generates a hypothetical gene tree (green).





To treat a set of gene sequences as being distributed across multiple populations, you incorporate them into a single matrix, then indicate to which population each sequence belongs. Three components must be established in your data file to do this:

1. A block of taxa representing the **gene sequences**. The sequences themselves do not need to be represented by a DNA matrix, although the matrix could of course be important for some analyses. For purposes of gene tree simulations, however, it is enough that each gene sequence be represented by a taxon in a taxa block.
2. A block of taxa representing the **populations** (or species). Each taxon in this block represents a different population.
3. A **taxa association block**, which is a special block of information that indicates how the taxa representing genes are associated with the taxa representing populations. It is this that indicates for each gene what population it belongs in.

Once these three components are established, you can ask to do calculations (e.g., gene tree simulations) using the genes in a way that pays attention to the populations in which each gene belongs. We will next describe how to set up these three components, then what calculations can be done.

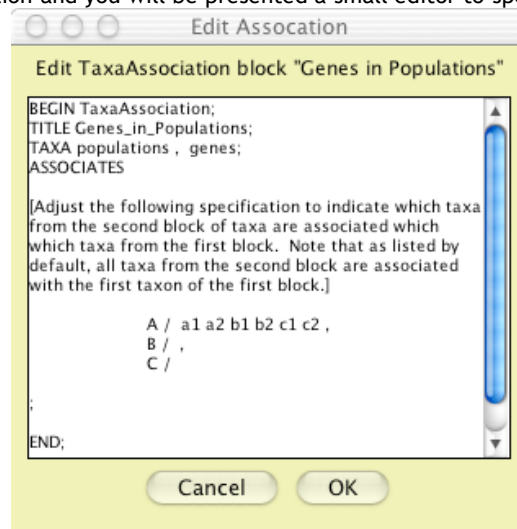
Establishing the association between genes and populations

Currently there are two ways to set up a data file with genes in multiple populations. Let's suppose you want to set up a file with six genes distributed among three populations (genes a1 and a2 in population A, b1 and b2 in population B, and c1 and c2 in population C).

Step-by-step method

The first method does it step-by-step, as follows:

1. Make a block of taxa representing the genes. This can be done by importing a data file with gene sequences, by creating a new file in Mesquite, or by selecting New Block of Taxa from the Taxa&Trees menu. Give this block of taxa a recognizable name, like "Genes". Give the genes (taxa) names in the List of Taxa window that will appear. In our example, the names are a1, a2, b1, b2, c1, and c2.
2. Make a block of taxa representing the populations or species containing the genes, and give it a recognizable name, such as "Populations". Give the populations (taxa) names in the List of Taxa window that will appear, e.g. A, B and C.
3. Select New Association... from the Taxa&Trees menu. You will be asked which will be the first block of taxa in the association. Although you can choose either genes or populations to be the first block, it will be easier for you to edit the taxa association if you choose populations as the first block. Then, name the association and you will be presented a small editor to specify the association:



This editor is crude; you edit directly in NEXUS file format. You can erase the instructional comment (make sure you delete everything from the "[" to the "]" inclusive). The key section to edit is the list of populations beside each of which is to be the list of its included genes. You can do this by Cut and Paste, cutting b1 and b2 and pasting them right after the "/" on the B line (in front of the ","), and cutting c1 and

c2 and pasting them right after the "/" on the C line, to yield:

```

Edit TaxaAssociation block "Genes in Populations"

BEGIN TaxaAssociation;
TITLE Genes_in_Populations;
TAXA populations , genes;
ASSOCIATES

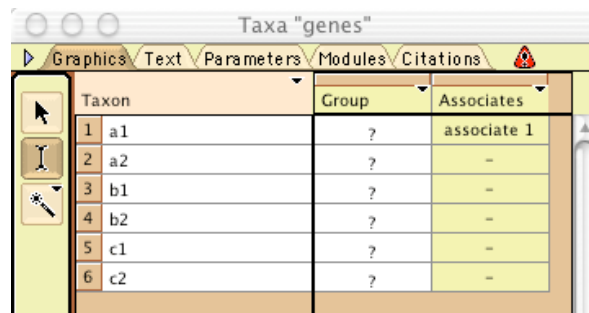
    A / a1 a2 ,
    B / b1 b2 ,
    C / c1 c2

;
END;
```

Click "OK" and your taxa association is ready to be used.

Automated method (using a macro)

A macro is available help you set up the gene-population association. It is called "Set up Associated Taxa Block" and is available in the Macros submenu of the Window menu. It assumes your file has one (and only one) block of taxa already made, that representing the genes. If you select this macro menu item, the second block of taxa representing populations will be created, the List of Taxa window for the genes will be shown, and a column labelled "Associates" will be shown, as follows:

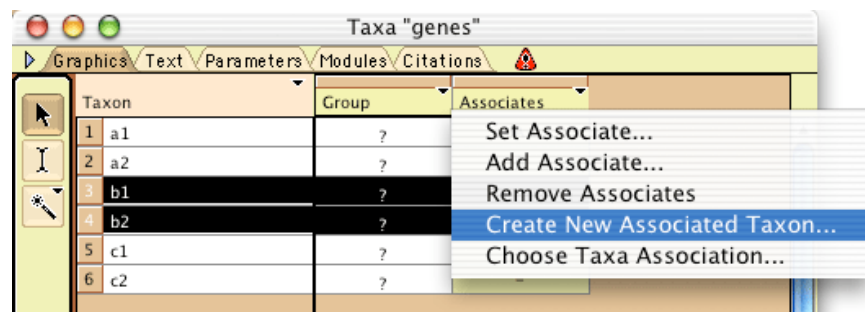


	Taxon	Group	Associates
1	a1	?	associate 1
2	a2	?	-
3	b1	?	-
4	b2	?	-
5	c1	?	-
6	c2	?	-

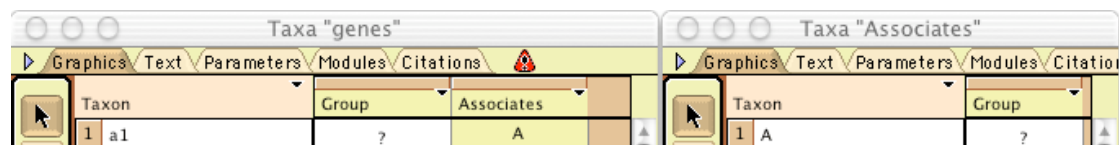
The column labelled "Associates" indicates what taxa in the second block (in this case populations) are associated. This macro automatically creates a single population called "associate 1" and indicates that the first gene belongs in it. [You will notice that the macro will show a window listing taxa in the new block, showing only the single taxon "associate 1".]. When the macro is done working, a dialog box will appear with some explanation.

You will probably want to rename "associate 1" to be the name of the population in which the first gene resides (in our case, "A"). You can do this in the List of Taxa window with the single row named "associate 1". Next, indicate what other genes belong with the first population by selecting their rows in the genes' List of Taxa window and touching on the column heading "Associates". Select "Set Associate" and choose the name of the first population.

You must now create the other populations and assign the genes to them. You can do that in several ways, but the quickest way is to select all of the genes that belong to the second population, and select Create New Associated Taxon in the menu that drops down when you touch the column heading "Associates":



In the dialog that appears indicate the name of this new population. Continue this until all genes are assigned to populations. You should end up with two List of Taxa windows that look something like this:



	Taxon	Group	Associates
1	a1	?	A

	Taxon	Group
1	A	?

2	a2	?	A
3	b1	?	B
4	b2	?	B
5	c1	?	C
6	c2	?	C

Editing already-created taxa associations

The instructions above indicate how to create two taxa blocks (genes, populations) and link them using a Taxa Association. After you have done this, you can modify the Taxa Association using either the direct editing of the TaxaAssociation block (to do this, select it from the "Edit Association" submenu of the Taxa&Trees menu) or via the Associates column of the List of Taxa window. To show the Associates column, make the List of Taxa window visible for the genes (selecting it under the List of Taxa submenu of the Taxa&Trees menu) then choose Associated Taxa under the Show Column submenu of the List menu.

Simulating coalescence within a population tree

Simulating gene trees evolving within a population tree is similar to simulating gene trees in a single population as described under Single Population. To be able to do it, however, you need to have already established an association between the gene taxa and populations, as described above. You also need to have a Tree Window open and showing a population tree, because the calculations need to be able to find a current population tree in which to perform the simulation.

If your data file is ready to go, you can generate and use gene trees simulated within the population tree in many contexts, such as a Tree Window or chart. You merely need to specify the source of trees to be Simulated Trees, and choose the simulator to be "Coalescence Contained within Current Tree". This is a secondary choice, and hence will be available under "Other Choices" in a menu or by selecting "Show Secondary Choices" in a dialog. Once you've selected Coalescence Contained, you will be asked to choose Effective Population Size.

The simulation starts at each extant population. Within each, the ancestry of the gene copies contained (as specified by the Taxa Association) is simulated by coalescence, going backward in time until the simulation arrives at the previous population divergence. These within-branch simulations use the same calculations and assumptions as the Single Population simulations (neutrality, panmixia). There is no migration among populations. The length of time allowed within this branch is the length of the branch, which is treated as the number of generations. (Thus, branch lengths of the population tree will typically be large, e.g., 1,000 to 1,000,000 or more.) The population size is determined by the chosen N_e , and is constant throughout the simulation unless modified by branch widths. Branch widths, which can be controlled by the Adjust Lineage Widths tool (the horizontal ruler) in the Tree Window, are treated as multipliers of the basic N_e . Thus, if the lineage width is unspecified or is 1.0, then the indicated N_e is used directly. If the lineage width of a branch is 0.5, then the population size along that branch is $0.5 \times$ (indicated N_e). Population fluctuations such as bottlenecks can be introduced explicitly along a single branch by inserting extra nodes within the branch using the Insert Nodes tool, and then varying the widths of the different segments independently. This is shown in the example file 08-fluctuating.nex.

By the time the simulation reaches a branch point, i.e. a population divergence, coalescence may have resulted in a single remaining ancestor of the sampled gene copies, or there may remain more than one ancestor. Whatever gene ancestors remain, they are united with the gene ancestors remaining in the sister population into the ancestral (pre-divergence) population. Coalescence then proceeds from there, moving backwards along the ancestral branch, and so on, until the root of the population tree is reached. Then the simulation continues in the root until only a single gene ancestor remains. The branch lengths in the resulting gene tree reflect the generations in which each coalescence occurred.

Reconstructing gene history within population history

If we are given a gene tree and a population tree, how can we interpret how the gene tree fits within the population tree? In what ancestral population did each gene divergence occur? Mesquite is currently able to make this reconstruction under only one assumption: that the only process occurring is lineage sorting (there is no migration among populations). Thus, the reconstruction reconciles the gene tree into the population tree (Page and Charleston, 1997) so as to minimize the depths of gene tree divergences (i.e., minimizing the implied incompleteness of lineage sorting).

This reconstruction is performed by Mesquite when visualizing gene trees within population trees using the Contained Associates tree drawing mode and when counting deep coalescences, both of which are described below. Three parameters determine how the reconstruction is done:

- **Treat Contained As Unrooted** – If enabled, then all possible rootings of the gene tree are tried to find that which minimizes incompleteness of lineage sorting. This is typically an appropriate setting for empirical gene trees (which are typically unrooted) but inappropriate for fitting simulated gene trees into the population history on which they were simulated, because the roots of these gene trees are known. (Default: disabled)
- **Contained Polytomies auto-resolve** – If enabled, then polytomies in the gene tree will be

automatically resolved into dichotomies so as to minimize incompleteness of lineage sorting. In the visualization of Contained Associates, such resolved areas are colored magenta. (Default: enabled)

- **Use Branch lengths of Contained tree** – If enabled, then the branch lengths of the contained (gene) tree will be respected in fitting into the population tree. Thus, if the gene tree's branches are long and the population tree's branches short, the gene tree will be interpreted as extending deep in time past the root of the species tree, even if this implies lineage sorting is more incomplete than it might otherwise need to be. If disabled, then the branch lengths of the gene tree are ignored in minimizing incompleteness of lineages sorting. For fitting simulated gene trees into the population history on which they were simulated, it is usually best to enable this option, for then the fit will reflect the actual history. (Default: enabled)

Visualizing gene history in population history

The visualization of green gene trees embedded with blue population trees shown elsewhere on this page is done by the Contained Associates tree drawer. This tree drawer can be used in various contexts where trees are drawn (e.g., the Tree Window), but it requires that the data file is already prepared with two blocks of taxa and their association. To ask for this visualization, have open a Tree Window showing trees of Populations (Contained Associates draws population trees). Select [Drawing>Tree Form>Other Choices](#) and then choose Contained Associates from the dialog box. You will be asked what gene trees to draw within the population tree. The gene trees could be from any of the usual sources: stored in the file, or simulated. You can choose to show gene trees simulated within the population tree itself at that moment.

Once Contained Associates is showing the tree, many of its controls will be in the Contained menu, not in the Drawing menu as usual. If you want the contained gene tree to appear in a separate window in addition to embedded in the population tree, select [Contained>Display Contained Tree](#).

Measuring fit between genes and populations

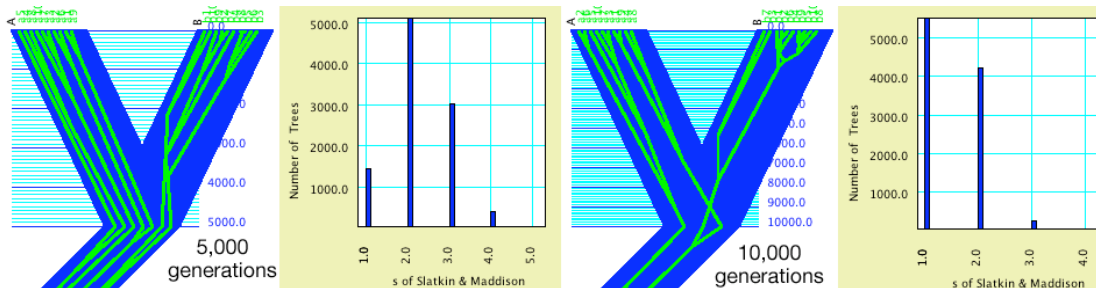
If the genes of a population do not form a clade in the gene tree (i.e., a monophyletic group) on the gene tree, then there is discordance between the gene tree and the population subdivision. To measure this discordance, Mesquite currently has two measures:

- ***s* of Slatkin & Maddison (1989)**. This measures discord between the gene tree and the subdivision into populations; it does not pay attention to a population tree. It treats the populations as a categorical character and counts the number of parsimony steps in this character on the gene tree; the more scattered on the gene tree are the genes from a population, the higher the *s* value. If the populations have been long separate and the only process causing this scatter is migration, then *s* can be interpreted as the minimum number of migration events between the populations. The *s* statistic can be calculated for gene trees whenever a Taxa Association is available indicating how the genes are associated with populations.
- **Deep coalescences of W. Maddison (1997)**. This measures the discordance between a gene tree and a population tree. It assumes that all discord between the population and gene trees is due to incomplete lineage sorting, and counts the number of extra gene lineages (beyond the minimum of 1) on each branch of the population tree summed over all population branches. The number of extra lineages is counted on the best fit of the gene tree into the population tree using the reconstruction methods described in the previous section. Deep coalescence can be calculated in various ways:
 - Deep Coalescences (gene tree) – This calculates deep coalescence from the gene tree's point of view. That is, it can be applied to a gene tree. It seeks a current population tree (for instance, on in a Tree Window) and measures the fit of the gene tree within that current population tree.
 - Deep Coalescences (species tree) – This calculates deep coalescence from the population (or species) tree's point of view. That is, it can be applied to a species tree. It seeks a gene tree from available tree sources (e.g., gene trees stored in the data file) and measures how well it fits within the species tree
 - Deep Coalescence Multiple Loci – This also calculates deep coalescence from the population (or species) tree's point of view, but for multiple gene trees simultaneously. It seeks a set of gene trees from available tree sources (e.g., gene trees stored in the data file) and sums deep coalescences for all of them. For instance, if you choose Stored Trees as the source of gene trees, all of the gene trees in a single stored tree block will be used. By summing deep coalescences for these gene trees, each is treated as if its descent was independent from the others, that is, an independent locus. You might expect each locus to be represented by a different block of taxa, but the Coalescence package of Mesquite cannot yet sum deep coalescences across taxa blocks. What if you include loci with different sample sizes in the different populations (or species), for instance locus P has 7 sequences from species A, 3 from B, 4 from C, while locus Q has 5 sequences from A, 3 from B and 6 from C? You can create a taxa block with sufficient taxa to accommodate both (7 genes contained in A, 3 in B, 6 in C) and then for each the gene trees representing the different loci exclude the extra genes as needed.

These measures assign a value to a tree, and thus are available as "Numbers for Trees". They can be calculated and displayed in various contexts, such as a histogram of values for trees, or in the Tree Legend, or as a column in the List of Trees window. They can also be used in tree searches.

Example: Effect of population divergence time on s

If you have reconstructed a gene tree from sampled sequences from two populations, you may want to use the degree of scrambling of the genes from the two populations as a measure of completeness of lineage sorting, and thus time since divergence. Although the s statistic was designed to measure gene flow, it might be used instead to measure time since divergence (assuming there is no ongoing gene flow). In the example below simulations are used to derive the expected s values under different divergence times. Ten genes are in each of two populations. A population tree with branch lengths of 5,000 and 10,000 are compared. These branch lengths are used as number of generations for the coalescence simulations, which here are using an effective population size of 10,000. The green-in-blue images of gene trees in population trees show the Contained Associates tree drawing mode of the Tree Window, with the gene trees simulated by Coalescence Contained within Current Tree. The charts are a separate calculation based on the same population trees; they are Histograms for Trees, in which the block of taxa for the trees are the genes, the value for the trees is the s statistic, and the gene trees are derived by simulations using Coalescence Contained within Current Tree. Note that with divergence 5,000 generations ago, s values of 3 are fairly common, whereas they are quite uncommon with divergence at 10,000 generations. By adjusting branch lengths, hypotheses can be tested and confidence limits derived.



Inferring the population or species tree

Mesquite can infer relationships of populations using contained gene trees or gene sequences, but currently its algorithms are relatively crude. Tree search and Cluster analysis are two options.

Tree search

Population trees may be inferred via a tree search that finds those population trees in which observed gene trees fit best (Maddison, 1997; Page and Charleston, 1997). Mesquite's tree search facility can be used to seek population trees that minimize deep coalescences (Maddison, 1997). Select Taxa&Trees>Make New Trees Block From>Other Choices and indicate Tree Search in the dialog box. Select populations as the taxa for the new trees block (the search will produce population trees). When asked for the criterion for the tree search, check the "Show Secondary Choices". You will see three choices for deep coalescences. Deep Coalescences (gene tree) is inappropriate because it assesses deep coalescence from the point of view of the gene tree, but your goal is to assess and choose species (population) trees. Thus, choose either Deep Coalescence (species tree) or Deep Coalescence Multiple Loci. Deep Coalescence (species tree) will ask you what gene tree to use as the basis for measuring deep coalescence within the candidate population trees. Deep Coalescence Multiple Loci will ask you what block of gene trees to use. More details on these criteria are given under [measuring fit](#).

Mesquite's tree search does not yet infer branch lengths, and thus the fit between gene trees and species trees is measured so as to ignore branch lengths.

Cluster analysis

Cluster analysis can be used to infer population trees by similarity of contained genes. Select Taxa&Trees>Make New Trees Block From>Other Choices and indicate Cluster Analysis in the dialog box. Indicate you want the taxa of the new trees block to be populations (as we are building a populations tree). For the measure of distance, indicate Distance of Contained Taxa (this a secondary choice), and then for the distance among contained taxa choose Uncorrected Distance or Patristic Distance (a secondary choice). "Uncorrected distance" counts the simple number of difference in gene sequences, and thus choosing this will yield a population tree that depends on a data matrix of the contained genes, but not on a gene tree. "Patristic Distance" measures distance along the branches of the gene tree, and thus requires a gene tree but no gene sequence matrix. If for the method to count distances among contained taxa you choose "Closest", and then "Single Linkage" as the cluster method, you will have an inference method similar to that implied by Takahata (1989): the similarity between two populations is judged by their most similar pair of gene sequences (not their average pairwise sequence divergence).

Simulating sampled gene sequences

Simulations can be used to generate gene sequences evolved under genetic drift and various models of mutation, either within a single population or within a history of diverging populations. To do this, the

gene trees simulated as described above (in a [single population](#) or [multiple populations](#)) are used as the basis, and mutations layered over top of the gene tree to yield a series of simulated sampled sequences. If done on a single gene tree, a gene sequence matrix results (each taxon a sampled gene copy; each character a site in the sequence). However, one can replicate this process automatically to produce many matrices, and thus obtain statistical distributions to test hypotheses. We introduce below some of the possible ways to generate sequences. For more details see the [Character Simulations](#) page, which explains how the Genesis package of Mesquite can be used to simulate nucleotide evolution.

To simulate sequence evolution you should first define a model of evolution as described [here](#). A key issue in simulating is using the scaling factor of the model to compensate for the units by which branch lengths are measured. Gene trees simulated by coalescence have branch lengths measured in generations, which may be in the thousands or millions, whereas most standard stochastic models expect trees whose branch lengths are much less than 10 for typical sequence divergences. For gene trees with lengths measured in generations, small scaling factors (e.g., less than 0.0001) should be used. We do not yet have recommendations as to exactly what scaling factor to use. We suggest you simulate a few matrices to find the scaling factor that gives you sequence divergences in the range desired.

Generating a single matrix of sequences

Gene tree already available in Tree Window

If a gene tree (simulated or otherwise) is shown in a Tree Window, you can simulate sequences simply by selecting **Characters>Make New Matrix From>Simulated Matrices on Current Tree**. (By "shown in a Tree Window" we don't mean shown as a thin green tree within the blue population tree in the Contained Associates drawing mode. We mean in a Tree Window dedicated to showing trees for the block of taxa corresponding to genes.) If your file has multiple taxa blocks you'll be asked for which you want a new matrix; indicate the taxa corresponding to genes. Indicate that you want Evolve DNA Characters. You will be asked to choose a model of evolution and a number of characters (i.e., sequence length). After it's done, the simulated matrix will be shown to you in a Character Matrix Editor window. If the sequences appear highly saturated (many changes) it may mean the scaling factor was improperly set.

Gene tree not in Tree Window

Select **Characters>Make New Matrix From>Other Choices...**, then indicate Simulated Matrices on Trees as the choice. Using this, Mesquite will get the gene tree on which to simulate sequences not from a Tree Window, but from some other available source of trees, such as gene trees stored in a trees block, or simulated at that moment by coalescence. After being asked for details about the model of evolution and number of characters, you will be asked to specify "Source of trees on which to simulate character evolution for matrices". Here you choose the source of gene trees. One possibility is to choose Simulated Trees then Coalescent Trees or (under secondary choices) Coalescence Contained within Current Tree, in which case the matrix will be simulated on a gene tree simulated by coalescence at that moment.

Generating a series of matrices

Multiple replicate sequence matrices can be generated and stored to files. This can allow you to perform a statistical test, for instance generating 100 sequence matrices under some hypothetical scenario, then examining them to see if their properties match those of an observed matrix.

On a single gene tree

To generate multiple sequence matrices evolved on a single gene tree, display the gene tree in a Tree Window. Then choose **Characters>Save Multiple Matrices>Simulated Matrices on Current Tree**. You will be asked to set up the simulation as for a single matrix (see above), except that you will also be asked to supply a base name for the file, the number of matrices, and the file format. If you choose "test" as the base name, 4 matrices, and the NEXUS file format, then four matrices will be simulated and written to the files test0.nex, test1.nex, test2.nex and test3.nex. You will be asked where to save the files.

Each matrix on a different gene tree

Matrices alone —To simulate a series of sequence matrices, each one evolved on a separate gene tree, select **Characters>Save Multiple Matrices>Other Choices...**, then select Simulated Matrices on Trees. Your choices will be similar to the preceding single tree case, except that instead of automatically choosing a gene tree in a tree window, it will use gene trees from the selected tree source. The first matrix will be simulated on the first gene tree from the tree source, the second from the second, and so on. Thus, if you choose Simulated Trees, Coalescent Trees as your tree source, then you will be generating a series of matrices, each simulated on a different gene tree simulated by coalescence within a population. These matrices represent a series of replicates of samples of gene sequences from a population evolving under drift with the specified model of mutation. You can similarly use Coalescence Contained within Current Tree to simulate genes evolving in a divergent population history.

Matrices plus batch files —The many sequence files generated by the preceding option can be analyzed by hand or by some other program. However, if you want Mesquite to help you analyze them automatically,

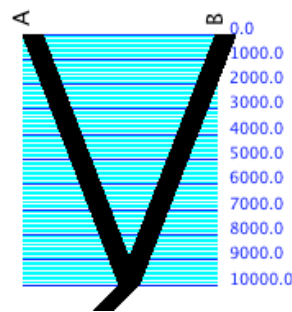
you can ask it to generate batch files that can script Mesquite or some other program to analyze the resulting files one after another, automatically. A batch file is simply a text file with instructions to a program; what to do, step by step. Whether you can do your desired analysis this way depends on whether the program you want to use to analyze the files can be scripted using batch files, and whether someone has designed a batch file template for the analysis.

To generate a series of matrices and corresponding batch files to analyze them, select [Analysis>Batch Architect>Export Matrices & Batch Files](#). You will be asked to specify how the matrices are to be generated, and then you will be presented with a dialog titled "Export Matrices & Batch Files". In this dialog you indicate the number of matrices to save, the base name for the files, and what batch file template to use. An example is given below, and details on the use of Export Matrices & Batch Files are given on the page on [Character Simulations](#).

Example: Multiple simulations of sequence samples

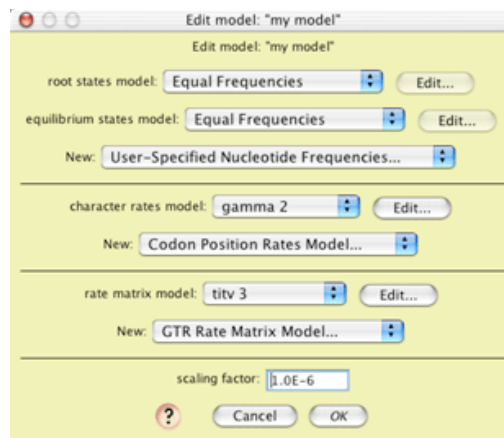
Suppose you had 20 gene copies sampled from two populations A and B and you reconstruct a gene tree. You notice that the copies from A and B do not form respective monophyletic groups, but rather are somewhat intermingled on the tree. You calculate the degree of intermingling using Slatkin & Maddison's s , and determine it to be 4. You want to know: what would be the probability of observing an s of 4 if the two populations had effective population sizes about the same as the number of generations since divergence, say 10,000. (Assume the populations have been completely isolated since divergence.) How can this probability be calculated?

First set up a data file with 20 genes associated with 2 populations, and display a tree window for populations showing a tree with divergence at 10,000 generations:



One way to get a quick answer is to select [Analysis>New Histogram for>Trees](#) and indicate you want trees representing genes, the value to calculate (under secondary choices) s of Slatkin and Maddison, and indicate you want Simulated Trees, with the tree simulator Coalescence Contained within Current Tree. This would simulate a series of gene trees within your proposed population history and plot their distribution of s values. However, these are the true gene trees simulated, and you don't know that your empirical gene tree is in fact correct. It would be better if you simulated not just gene tree evolution, but gene tree reconstruction also, so that you would be able to compare simulated *reconstructed* gene trees with your empirical *reconstructed* gene tree. Thus, we will simulate the gene trees, simulate sequence evolution on them, and take those sequences and attempt to reconstruct gene trees from them.

The procedure is as follows. First, set up the file to have the two taxa blocks (genes, populations) and the association between them. Display a tree window for the populations and set it to match your hypothetical scenario of population history. Our population history has two populations diverging 10,000 generations ago, with effective population size of 10,000. Build a model of sequence evolution. For instance, we built one like this:

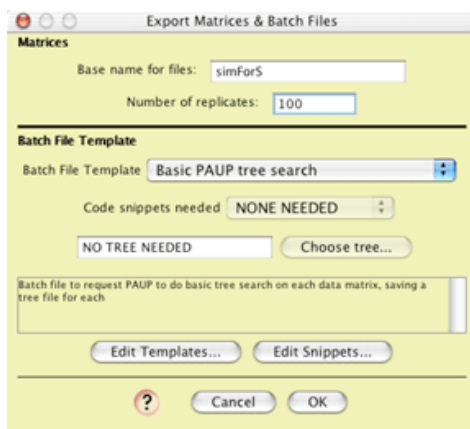


The scaling factor (here, 1.0E-6 which is 0.000001) was chosen to yield fairly low sequence divergences

under the expectation that the branch lengths on the gene tree could be as long as 10,000 to 50,000 (generations). We tested it by simulating a few matrices to see that the divergences were as desired.

Then choose **Analysis>Batch Architect>Export Matrices & Batch Files**, indicating to save matrices for the genes. The matrices to be exported come from **Simulated Matrices on Trees** (a secondary choice). Choose **Evolve DNA Characters**, your model, and 1000 characters to indicate how each matrix is to be simulated. For the trees on which to simulate, indicate **Simulated Trees**, then **Coalescence Contained within Current Tree** (a secondary choice). Indicate an N_e of 10,000.

You will be shown the **Matrices & Batch Files** dialog:

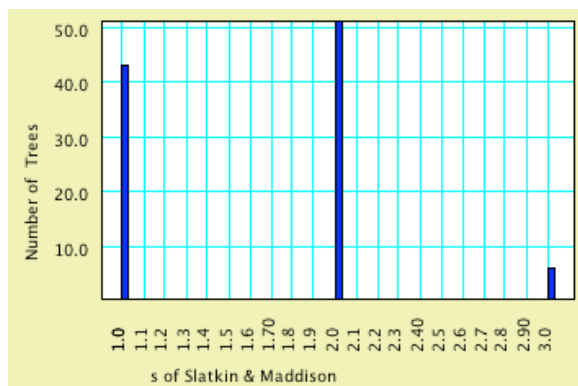


Choose a template that will reconstruct the trees for you from the matrices. We will assume for the rest of this example that the template "Basic PAUP Tree Search" was chosen. Indicate how many matrices to make, and the base name to give to the files (here, "simForS"). Generating the matrices may take a while. Produced will be a series of files (here, "simForS0.nex", "simForS1.nex", etc.) with the matrices, and the files **TreeFileList**, **[base name]BasisTrees.nex**, and **commands.nex**. **TreeFileList** is a simple text file listing tree file names that will be produced by PAUP*, in this case **simForS0.trees**, etc. **[base name]BasisTrees.nex** (in this example, its name will be **simForSBasisTrees.nex**) stores all of the gene trees simulated and used to generate the matrices. These are saved to document what was done in the simulation. The file **commands.nex** is the batch file that tells PAUP* to execute each of the data files and for each infer trees. Start PAUP* and ask it to execute **commands.nex**.

PAUP* should produce the following files, assuming the base name you choose is "basename":

- **basename0.trees**, **basename1.trees**, etc. – trees reconstructed by PAUP from each of the matrices
- **CBbasename** – the consensus trees from each of the tree searches, stored as a series of distinct tree blocks in a single file
- **consensus.trees** – the consensus trees fused into a single tree block
- **ConsCons.trees** – the consensus of the consensus trees from all the matrices. (In this example, this file is not very useful and will likely contained fully unresolved tree.

The key file for us is **consensus.trees**, because it contains the 100 trees reconstructed, one for each matrix. Back in Mesquite, ask to **Link or Include** this file. Then select **Analysis>New Histogram for>Trees** and indicate you want trees representing genes, the value to calculate (under secondary choices) s of Slatkin and Maddison, and indicate you want **Stored Trees**. These **Stored Trees** will be those reconstructed from the simulated matrices. The chart may look like this:



As you can see, it appears unlikely that you would reconstruct a gene tree giving an s value of 4 under the scenario of population history.

References

Maddison, W.P. 1997. Gene trees in species trees. *Systematic Biology* 46:523-536.

Slatkin, M. and W. P. Maddison. 1989. A cladistic measure of gene flow inferred from the phylogeny of alleles. *Genetics* 123: 603-613.

Takahata, N. 1989. Gene genealogy in three related populations: Consistency probability between gene and population trees. *Genetics* 122:957-966.

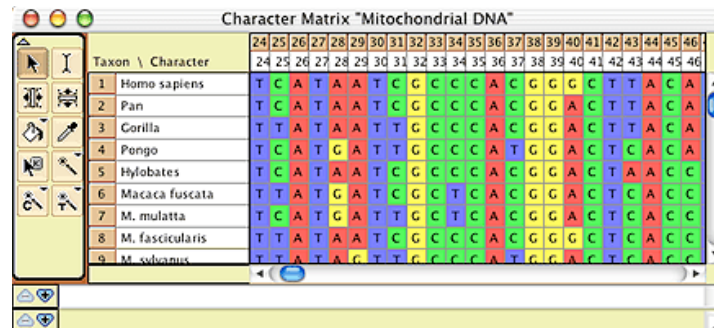
Page, R.D.M and M.A. Charleston. 1997. From gene to organismal phylogeny: Reconciled trees and the gene tree species tree problem. *Molecular phylogenetics and evolution*. 7:231-240.

Molecular data

Molecular data (DNA or protein sequences) can be edited, manipulated, simulated and analyzed in various ways in Mesquite. Most of the features discussed elsewhere concerning editing and analysis of general categorical data also apply to molecular data; here we focus on features specifically designed for sequence data.

Contents

- [Editing molecular data](#)
- [Simulating DNA sequence evolution](#)
- [Statistics for DNA sequences](#)
- [Statistics for Protein sequences](#)
- [Visualizing tertiary structure](#)
- [Sequence Data within populations](#)
- [Reconstructing ancestral states](#)



The screenshot shows the 'Character Matrix "Mitochondrial DNA"' window in Mesquite. It displays a table with taxa as rows and characters as columns. The taxa listed are Homo sapiens, Pan, Gorilla, Pongo, Hylobates, Macaca fuscata, M. mulatta, M. fascicularis, and M. melanurus. The characters are numbered from 24 to 46. The matrix contains nucleotide data (A, C, G, T) for each taxon across the characters. The interface includes a toolbar on the left with icons for various editing and analysis functions.

Taxon \ Character	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46
1 Homo sapiens	T	C	A	T	A	T	C	G	C	C	C	A	C	G	G	C	T	T	A	C	A		
2 Pan	T	C	A	T	A	T	C	G	C	C	C	A	C	G	G	A	C	T	T	A	C	A	
3 Gorilla	T	T	A	T	A	T	T	G	C	C	C	A	C	G	G	A	C	T	T	A	C	A	
4 Pongo	T	C	A	T	G	A	T	T	G	C	C	A	T	G	G	A	C	T	T	A	C	A	
5 Hylobates	T	C	A	T	A	T	C	G	C	C	C	A	C	G	G	A	C	T	T	A	C	C	
6 Macaca fuscata	T	T	A	T	G	A	T	C	G	C	T	C	A	C	G	G	A	C	T	T	A	C	C
7 M. mulatta	T	C	A	T	G	A	T	T	G	C	T	C	A	C	G	G	A	C	T	T	A	C	C
8 M. fascicularis	T	T	A	T	A	T	C	G	C	C	C	A	C	G	G	G	C	T	T	A	C	C	C
9 M. melanurus	T	T	A	T	A	T	C	G	C	C	C	A	C	G	G	G	C	T	T	A	C	C	C

Editing molecular data

Molecular data can be imported from files of NBRF format, PHYLIP format, and simple table format. It can also be exported to these formats.

The [Character Matrix Editor](#) can be used to edit a molecular sequence matrix. Standard ambiguity codes are allowed.

The following can be applied to all or the selected portions of a molecular sequence matrix in the Character Matrix Editor. These are available under the Alter/Transform submenu of the Matrix menu:

- **Nucleotide complement** (DNA matrix only) – enters the complementary sequence into the selected cells
- **Reverse sequence** – reverses the order of contiguously selected blocks of sequence

Other options may appear; see the page on [characters](#) for standard choices in this submenu. You can also apply the other editing tools described for [character matrices](#).

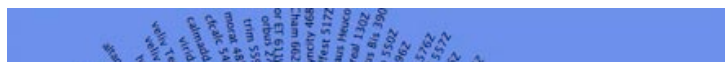
Simulating DNA sequence evolution

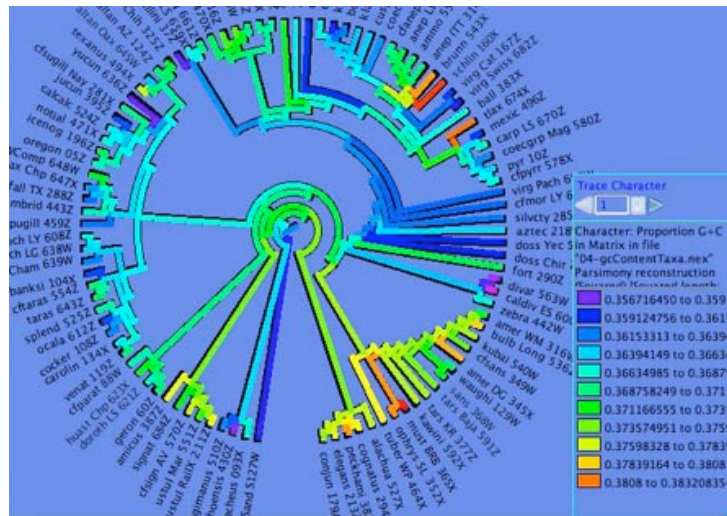
DNA sequence evolution can be simulated to build statistical tests, for instance via parametric bootstrapping. See the page on [simulating DNA sequences](#).

Statistics for DNA sequences

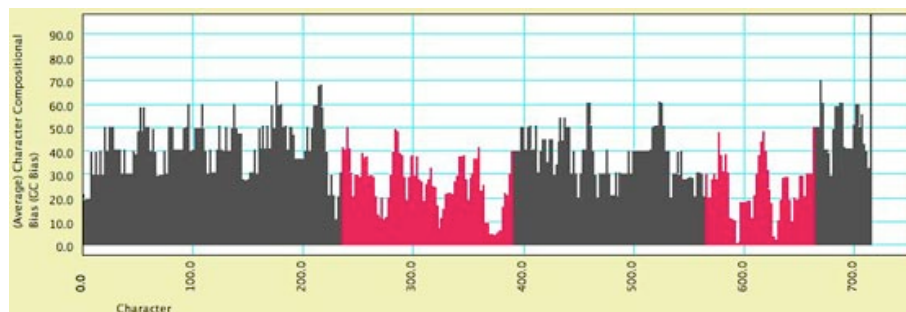
Calculations for categorical characters in general can be applied to DNA sequences. For example, [Parsimony calculations](#) can be made for DNA sequences, as can basic descriptive statistics such as the percent of a sequence or character that is missing data or gaps. In addition, there are several modules specifically designed for DNA data, illustrated by examples in Mesquite_Folder/examples/Molecular. These calculate compositional bias:

- **ACGT Compositional Bias** – This module supplies the compositional bias of taxa, measured over the taxon's sequence. The bias is treated as a continuous character, and thus can be used wherever characters are used, as for instance in the reconstruction of the evolution of compositional bias as shown in the image below. It can return either the proportion G+C, or separately A, C, G, and T proportions.

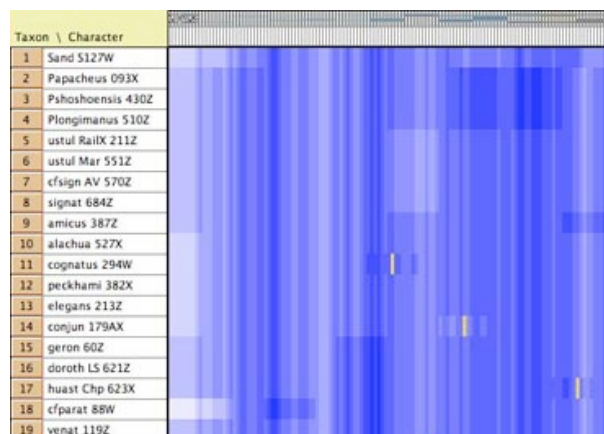




- **Character Compositional Bias** – This module supplies the compositional bias for characters. It calculates the percent of taxa with particular nucleotides (GC bias, or individual frequency of A, C, G or T) for a character. The image below shows a moving window analysis of compositional bias along a sequence; the instructions for generating the chart are given [here](#).

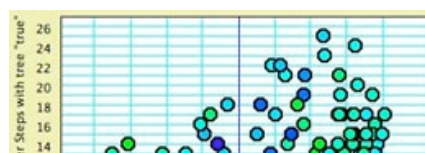


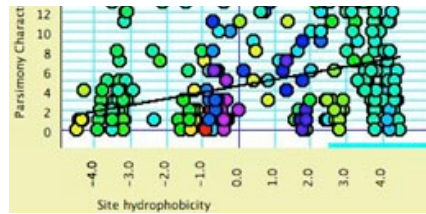
- **GC bias coloring of matrices** – The cells of the Character Matrix Editor may be colored according to a moving window of GC bias along the sequence, as shown below, by selecting **Matrix>Color Cells>Color By Value**, then once shown the colors can be smoothed by a moving window analysis by selecting **Matrix>Moving Window (for colors)**.



Statistics for Protein Data

- **Site hydrophobicity** – This module supplies the average amino acid hydrophobicity, averaged across taxa, for each site. It can be used in charts, for instance to see the relationship between a phylogenetic statistic for the site (character) and its average hydrophobicity. This [chart](#), for example, shows parsimony character steps as a function of hydrophobicity:



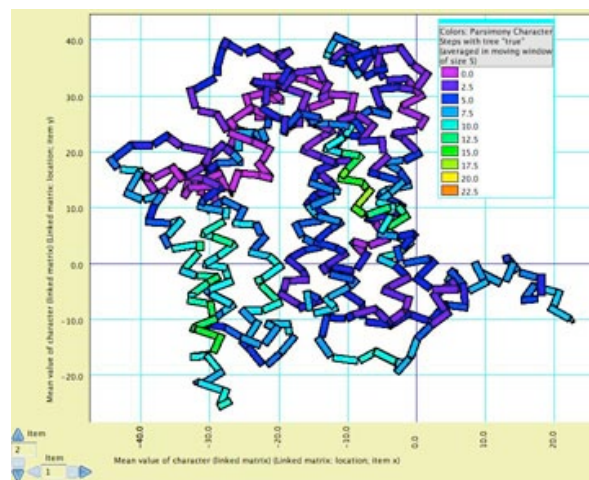


- **Amino Acid hydrophobicity** – The cells of the Character Matrix Editor may be colored according to a moving window of hydrophobicity along the sequence, as shown below, by selecting Matrix>Color Cells>Color By Value, then once shown the colors can be smoothed by a moving window analysis by selecting Matrix>Moving Window (for colors).

Taxon \ Character	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	96	97	98	99	100
1 Lamprey	S	H	Q	P	S	I	I	R	K	T	H	P	L	S	L	G	N	S	M	L	V	D	L	P	S	N	I	S	A	W	N	F	G	S	L	S	L	C	L	I	L	Q	T	I	T	G	L	L	A	M	H	Y	T	A	N	T	E	L	A	F	S	S	V	M	H	I	C	R																																
2 Alligator	M	T	H	Q	L	R	K	S	H	P	I	K	L	I	N	R	S	L	I	D	L	P	T	S	N	I	S	A	W	N	F	G	S	L	L	G	L	T	L	I	Q	I	L	T	G	F	L	M	M	H	F	S	S	D	T	L	A	F	S	S	V	Y	T	S	R																																			
3 crow	M	G	L	N	L	R	K	N	H	P	L	L	K	I	I	N	N	S	L	I	D	L	P	T	S	N	I	S	A	W	N	F	G	S	L	L	G	L	C	L	I	M	Q	I	T	G	L	L	A	M	H	Y	T	A	D	T	S	L	A	F	A	S	V	A	H	M	C	R																																
4 chicken	M	A	P	N	I	R	K	S	H	P	L	K	M	I	I	N	N	S	L	I	D	L	P	A	P	S	N	I	S	A	W	N	F	G	S	L	L	A	V	E	L	M	T	Q	I	L	T	G	L	L	A	M	H	Y	T	A	D	T	S	L	A	F	S	S	V	A	H	T	C	R																														
5 ostrich	M	A	P	N	I	R	K	S	H	P	L	K	I	I	N	N	S	L	I	D	L	P	S	P	S	N	I	S	A	W	N	F	G	S	L	L	G	I	C	L	I	T	Q	I	L	T	G	L	L	A	M	H	Y	T	A	D	T	L	A	F	S	S	V	A	H	T	C	R																																
6 Chrysem	M	T	M	N	H	R	K	T	H	P	L	T	K	I	I	N	N	S	F	I	D	L	P	S	P	S	N	I	S	A	W	N	F	G	S	L	L	G	T	C	L	I	L	Q	T	I	T	G	I	F	L	A	M	H	Y	S	P	D	I	S	L	A	F	S	S	V	A	H	I	T	R																													
7 Pelomed	M	G	T	L	H	L	K	Q	N	P	L	L	K	I	T	N	K	S	L	I	N	L	P	S	P	S	N	I	S	A	W	N	F	G	S	L	L	G	M	C	L	I	L	Q	I	T	T	G	I	F	L	A	M	H	Y	T	P	N	I	T	A	F	S	S	V	A	H	I	T	R																														
8 Blue Whale	M	T	N	I	R	K	T	H	P	L	M	K	I	I	N	D	A	F	I	D	L	P	T	S	N	I	S	S	W	N	F	G	S	L	L	G	L	C	L	I	V	Q	I	L	T	G	L	F	L	A	M	H	Y	T	P	D	T	M	T	A	F	S	S	V	T	H	I	C	R																															
9 Fin whale	M	T	N	I	R	K	T	H	P	L	M	K	I	V	N	D	A	F	V	D	L	P	T	S	N	I	S	S	W	N	F	G	S	L	L	G	L	C	L	I	M	Q	I	L	T	G	L	F	L	A	M	H	Y	T	P	D	T	T	A	F	S	S	V	T	H	I	C	R																																
10 Hippo	M	T	N	I	R	K	S	H	P	L	M	K	I	I	N	D	A	F	V	D	L	P	A	P	S	N	I	S	S	W	N	F	G	S	L	L	G	V	C	L	I	L	Q	I	L	T	G	L	F	L	A	M	H	Y	T	P	D	T	L	A	F	S	S	V	T	H	I	C	R																															
11 Pig	M	T	N	I	R	K	S	H	P	L	M	K	I	I	N	N	A	F	I	D	L	P	A	P	S	N	I	S	S	W	N	F	G	S	L	L	G	I	C	L	I	L	Q	I	L	T	G	L	F	L	A	M	H	Y	T	S	D	T	T	A	F	S	S	V	T	H	I	C	R																															
12 sheep	M	I	N	I	R	K	T	H	P	L	M	K	I	V	N	N	A	F	I	D	L	P	A	P	S	N	I	S	S	W	N	F	G	S	L	L	G	I	C	L	I	L	Q	I	L	T	G	L	F	L	A	M	H	Y	T	P	D	T	T	A	F	S	S	V	T	H	I	C	R																															
13 cow	M	T	N	I	R	K	S	H	P	L	M	K	I	V	N	N	A	F	I	D	L	P	A	P	S	N	I	S	S	W	N	F	G	S	L	L	G	I	C	L	I	L	Q	I	L	T	G	L	F	L	A	M	H	Y	T	S	D	T	T	A	F	S	S	V	T	H	I	C	R																															
14 white rhino	M	T	N	I	R	K	S	H	P	L	K	I	I	N	H	S	F	I	D	L	P	T	S	N	I	S	A	W	N	F	G	S	L	L	G	I	C	L	I	L	Q	I	L	T	G	L	F	L	A	M	H	Y	T	P	D	T	M	T	A	F	S	S	V	A	H	I	C	R																																
15 Black Rhino	M	T	N	I	R	K	S	H	P	L	V	K	I	I	N	H	S	F	I	D	L	P	T	S	N	I	S	S	W	N	F	G	S	L	L	G	I	C	L	I	L	Q	I	L	T	G	L	F	L	A	M	H	Y	T	P	D	T	T	A	F	S	S	V	T	H	I	C	R																																

Visualizing tertiary structure

Although there are not yet dedicated windows for visualizing phylogenetic statistics in the context of molecular structure, features have been added to the Scattergram chart to allow it to be adapted for this purpose. For instance, in this image cytochrome B is shown, with the amino acids colored according to a simple phylogenetic statistic: the number of parsimony steps on a phylogeny. The colors are smoothed by a moving window, and show that several coils of the molecule, a few at the left and one deep at the right, evolve more rapidly than others. This example is illustrated in the data file at Mesquite_Folder/examples/Molecular/06-cytochromeB.nex



To build such a chart, begin with a file with a matrix of protein sequences. The procedure is also described in the example files 08-cytochromeBlinked.nex and 09-cytochromeBscatter.nex.

- Select **New Linked Matrix** from the Characters menu. When a matrix is made to be linked to a second matrix, the two matrices are constrained to have the same number of characters.
- Indicate that you want the linked matrix to be a Continuous matrix, and link it to your protein matrix. Then, turn it into a three dimensional matrix (Taxa X Characters X Coordinates [x, y and z]) by using **Add Item** and **Rename Item** in the Utilities submenu of the Matrix menu of the Character Matrix Editor. The x,y,z coordinates could be added for all taxa if known, but otherwise only one taxon needs to be filled out (because we will use the average x,y,z coordinates for the amino acids).
- Once the linked matrix of xyz amino acid positions is entered, select Analysis>New Scattergram for> Characters. Indicate you want the scattergram to be for **Stored Characters**, and indicate **Same** value for the two axes. In the dialog box "Values for axes", choose **Mean Value of Character (Linked Matrix)**. In response to "Use characters from which matrix? (for Character Source)" choose the protein sequence matrix as the matrix to be used. This will plot the sites (amino acids, characters)

- in their correct places, but as a series of round spots.
- To change the appearance of the plot, select **Join the Dots** in the Special Effects submenu of the Scattergram menu. Then select **Thick Joints**, deselect **Show Dots**, deselect **Join First to Last**, and set the marker size larger (e.g., 8). This will result in a plot as shown above, but without the colors.
- Next, choose **Color by Third Value** from the Colors menu and choose the value by which to color the amino acids. For parsimony steps, for instance, choose **Character Value with current tree**.
- Finally, to use a moving window to smooth the colors, select **Moving Window for Colors** from the Colors menu and indicate the window size (e.g., 5).

Sequence data within populations

See the page on [population genetics](#).

Reconstructing ancestral states

Ancestral states of continuous characters can be reconstructed as described in the page on [reconstructing ancestral states](#). Likelihood methods are not yet available for molecular characters.

Continuous Characters

Continuous characters (e.g., with values 1.21, 5.68, and so on) can be edited, manipulated, simulated and analyzed in various ways in Mesquite. Below is a brief outline of these features, some of which come from the standard packages of Mesquite, others of which come from the built-in Rhetenor package (by Dyreson and Maddison) and the separately-available [PDAP](#) package (by Midford, Garland & Maddison). Many of these calculations are more thoroughly illustrated in the example files under

[Mesquite Folder/examples/Basic Examples/continuous/](#), and under [Mesquite Folder/examples/Multivariate Continuous/](#)

Contents

- [Editing continuous data](#)
- [Reconstructing ancestral states](#)
- [Plotting trees](#)
- [Simulating character evolution](#)
- [Ordinations](#)
- [Tree reconstruction](#)
- [Felsenstein's independent contrasts](#)
- [Geometric morphometrics](#)

Editing continuous data

Continuous data can be imported from tab-delimited tables in ASCII text files, or entered into the spreadsheet editor (Character Matrix Editor). Values entered can be negative or positive, and include exponential notation (e.g., "1.3e-6").

A continuous data matrix can have an extra dimension, in that the entry for each cell of the matrix (character state in a taxon) can have more than one number. These separate numbers are called "items", and thus a character matrix can be described as having three dimensions, characters X taxa X items. The first item could be the mean; the second the variance. Or, there could be 3 items, x, y and z, representing coordinates of a landmark in space. To manage items, use the Utilities submenu of the Matrix menu.

The following can be applied to all or the selected portions of a continuous matrix in the Character Matrix Editor. These are available under the Alter/Transform submenu of the Matrix menu:

- Fill – fills the cells with the current "paint" state
- Standardize – transforms the characters to have mean 0 and variance 1.
- Random fill – fills the cells with randomly generated states, with given mean and variance.
- Add random noise – adds random noise to the entries.
- Add constant – adds a specified constant to all entries.
- Multiply constant – multiplies all entries by a specified constant.

Other options may appear. You can also apply the other editing tools described for [character matrices](#).

Reconstructing ancestral states

Ancestral states of continuous characters can be reconstructed as described in the page on [reconstructing ancestral states](#).

Plotting trees

Trees can be mapped or plotted into a character space as described in the page on [processes of character evolution](#).

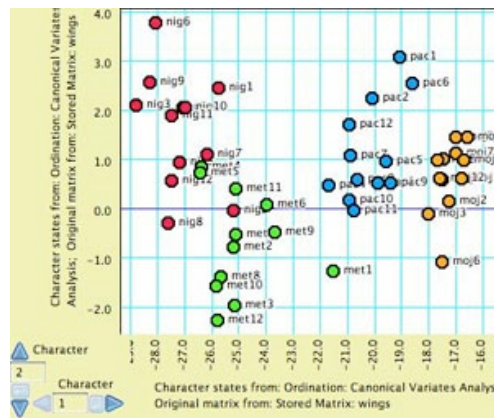
Simulating character evolution

Evolution of continuous characters can be simulated by selecting Simulated Characters or Simulated Matrices, and choosing Evolve Continuous Characters. You will get to choose a model, which will be used to simulate evolution on the tree. There is one default model, a Brownian motion model with rate parameter of 1.0. You can create alternative models (e.g. other Brownian motion models) by selecting New Character Model in the Characters menu.

Ordinations

Where matrices of continuous characters are used, for instance in plotting trees or in Taxa Scattergrams, it is possible to instead use characters representing the modified axes obtained by ordinations such as

Principal Components Analysis using modules in the built-in Rhetenor package. For instance, the following Taxa Scattergram shows the results of a Canonical Variates Analysis:



How to set up this plot is explained on the page on [Charts](#).

To use ordinations, simply select Characters from Ordinations or Matrices from Ordinations wherever you might otherwise select Stored Characters or Stored Matrices. There are several options for ordinations:

- Principal Components Analysis
- Canonical Variates analysis – This requires a Taxa Partition to exist to indicate groups of taxa.
- Among-group PCA
- Within-group PCA
- Evolutionary PCA (similar to PCA but tree-based)

The "Multivariate Continuous" example files illustrate the use of these methods.

Tree reconstruction

The tree search facility (available under [Trees&Taxa>Make New Trees Block from>Tree Search](#)) allows one to search for trees minimizing treelength as calculated by linear or squared change parsimony for continuous characters. It should be noted, however, that the current Tree Search facilities in Mesquite do not adjust branch lengths. The squared change parsimony algorithms by default weight by branch length. Thus, the search is done effectively under the constraint that all trees have branch lengths of 1.0.

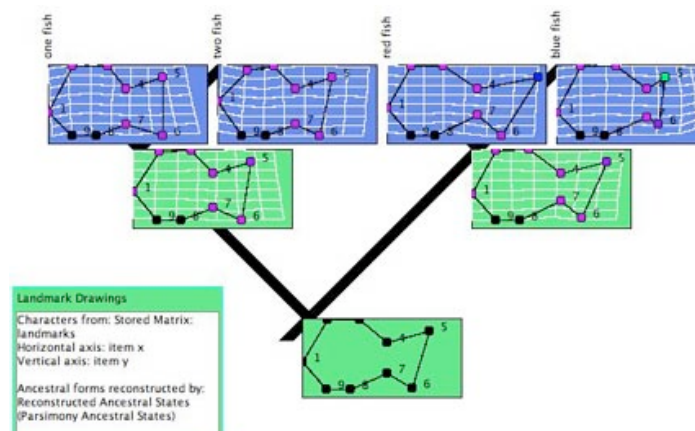
Felsenstein's independent contrasts

Analyses of character correlations can be done by the separately-available [PDAP](#) package. This is described briefly [here](#).

Geometric morphometrics

Landmark data can be entered in Mesquite as a continuous matrix with multiple items. Each character is a landmark, and each item is a dimension of the landmarks' coordinates. Thus, for two dimensional landmarks, the matrix could have the items "x" and "y".

Mesquite cannot yet perform Procrustes analyses to bring landmarks into a common scaling and alignment across taxa, but given that the data is already so prepared, the Landmark Drawings module of the Rhetenor package can reconstruct ancestral forms as shown below:



The algorithm used for the reconstruction is squared change parsimony.

Studies using Mesquite

- Study 1: [Testing monophyly of a group of beetles](#)
 - Study 2: [Are strepsipterans related to flies? Exploring long branch attraction](#)
-

Testing monophyly of a group of beetles

David R. Maddison

The question

There is an enigmatic group of terrestrial beetles called the Trachypachidae (left, below). These had traditionally been considered to be related to other terrestrial beetles in the suborder Adephaga, but later analyses of morphological data suggested (Bell, 1966; Hammond, 1979; Roughley, 1981; Ward, 1979) that they were instead closely related to some water beetles, the Dytiscoidea (right, below).



A terrestrial trachypachid (*Trachypachus holmbergi*, left) and an aquatic dytiscoid (*Hydrocanthus* sp., right).

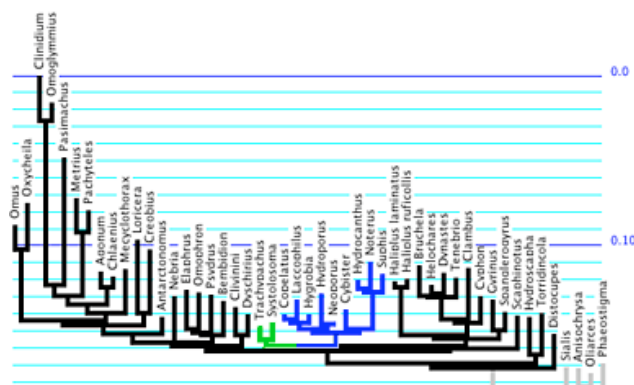
Shull et al. (2001) report sequence data of the 18S rRNA gene suggesting that trachypachids are not related to dytiscoids, but are instead related to some other terrestrial beetles. For example, the most parsimonious trees in one analysis have trachypachids with other terrestrial adephagans, and not with dytiscoids. Forcing trachypachids with dytiscoids increases the treelength by 9 steps.

However, given the vagaries of the evolutionary process and phylogenetic inference, it is possible that even if trachypachids and dytiscoids are truly related, one might incorrectly infer that they are not, just by chance or because of difficulties such as long branch attraction (Felsenstein, 1978; Hendy & Penny, 1989; Huelsenbeck, 1997). A statistical test could help us determine whether or not a difference of 9 steps between trees with trachypachids placed with dytiscoids and unconstrained trees is expected under the hypothesis that trachypachids and dytiscoids are related.

A statistical test

Is 9 steps a significant difference in treelength in this context? Can we reject the monophyly of trachypachids plus dytiscoids?

To test the hypotheses that trachypachids are related to dytiscoids, we first need to flesh out its details of the hypothesis. It needs to be detailed enough to allow us to predict the observations we would expect to see if this hypothesis were true. First of all, we will need a detailed phylogenetic hypothesis. We could find the tree of highest likelihood for 18S rDNA under the constraint that trachypachids are with dytiscoids; for one matrix, this tree is as shown below, with trachypachids marked in green and dytiscoids in blue.



Branch lengths of this tree and parameter values for a GTR + Gamma + Proportion Invariant (GTR+G+I) model of evolution are estimated by maximum likelihood using the 18S rDNA data. We thus have built a model that contains our best guess of the nature of evolution presuming that trachypachids and dytiscoids form a clade; the details of the model were established using the 18S rDNA data.

We can now ask: if evolution occurred under a GTR+G+I model with the parameter values as inferred, up the tree shown above (with trachypachids and dytiscoids forming a clade), then what would we expect the difference in treelength to be between the most parsimonious trees constrained to have trachypachids with dytiscoids and the most parsimonious unconstrained trees? Would that difference in general be similar to the observed value, 9? Or is 9 an unexpected value? Specifically, is 9 a value that we would expect to observe less than 0.05 of the time? If so, then we could reject the hypothesis that trachypachids are related to dytiscoids.

This test is an example of the use of parametric bootstrapping (Huelsenbeck et al., 1995; Swofford et al., 1996; Goldman et al., 2000). It is almost identical to the monophyly test proposed by Huelsenbeck et al. (1996), except that they use a difference in likelihood rather than treelength as their test statistic. Treelength is used here for pedagogical reasons as it allows the reader to conduct the test quickly; the methods described below could easily be modified to use likelihood values instead.

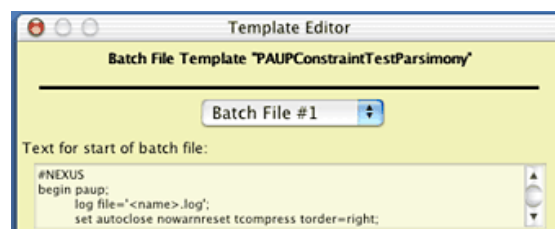
Building the statistical test

To conduct this test, in Mesquite open the example file "study001.nex", in the folder Mesquite_Folder/docs/mesquite/studies/study001/. This file contains a matrix of 18S rDNA, and the tree of highest likelihood found in which trachypachids are with dytiscids, with branch lengths inferred from the data. It also contains a GTR+G+I model with parameters inferred by PAUP* using maximum likelihood and 18S rDNA. We will ask Mesquite to simulate 100 matrices, and then (using the batch file Mesquite produces) ask PAUP* to find the most parsimonious trees with trachypachids constrained to be with dytiscoids and the most parsimonious unconstrained trees, and write their treelengths to a scorefile. Mesquite will then read in the results and calculate the distribution of treelength differences, allowing us to determine if the observed value of 9 is unusual.

The simulations can be done by choosing (Tree Window) Analysis > Batch Architect > Export Matrices & Batch Files.... After selecting "Simulated Matrices on Current Tree" and "Evolve DNA Characters", choose the GTR+G+I model. You will then be presented with the Export Matrices & Batch Files dialog box. Give a base name for the matrices of "TDTest", and do 100 replicates. We will use the template "PAUPConstraintTestParsimony":



If you wish to look at the contents of this template, touch on "Edit Templates", and in the Template Manager, select "PAUPConstraintTestParsimony", and press "View". (As this template is built-in, you can't edit it, only view it.) You will see some elements of the batch files that will be produced.



Repeated text, written for each matrix:

```
execute "<name><number>.nex";
set criterion=parsimony;
constraint theConstraint= <tree>;
hs enforce converse constraint=theConstraint;
pscore 1/scorefile="<name>Score.scr" append;
```

Text for end of batch file:

```
log stop;
end;
```

Batch File Name: PaupCommands.nex

Format of Matrices: NEXUS file

Explanation of template:

This template produces a batch file to request PAUP to find the most parsimonious trees that match a constraint, and the most parsimonious ones unconstrained, for each matrix. It asks PAUP to write the results to a score file.

? OK

The complexity of this template needn't be of concern. If you are interested, details of how it works are presented in "[Design of Batch Templates](#)". Just press "OK" to get back to the Template Manager, and then press "Done".

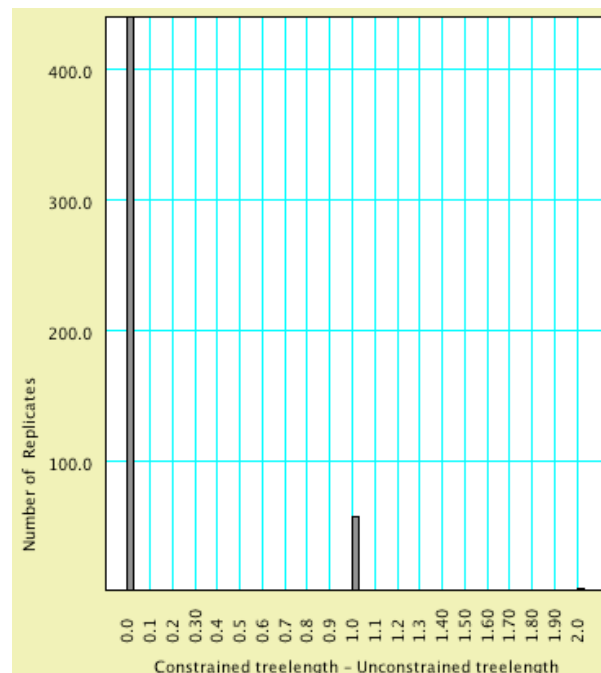
You are almost ready to have Mesquite simulate the matrices. However, the template we are using requires that we specify a tree to use as a constraint tree. In this case, the constraint tree is one with trachypachids with dytiscoids, but with no other structure. A tree like this is stored in the file under the name "TD constraint". To choose it, touch the "Choose Tree" button, and select "TD constraint".

All the options have now been chosen. Press the "OK" button in the Export Matrices and Batch Files dialog box to start the simulation. You will be asked for a location to save the files, and be given some messages. When you are asked for the number of characters, Mesquite recognizes that you are calculating some elements of the model using an existing matrix (in particular, the frequencies of A, C, G, and T), and for this reason it gives as the default number of characters the number in the original matrix. That's the number we want to use in this case, as we want the simulation to be as realistic as possible.

Once the simulations are all done, then go into PAUP*, and execute the file "PaupCommands.nex". When PAUP* completes its analyses, go back to Mesquite, and choose (Tree Window) Analysis > Batch Architect > Show Results via Instruction File.... Choose the file "**MesquiteInstructions**", in the same folder as the simulated matrices, and then choose the results file (which should be called "TDTestScore.scr",). Mesquite should then show you a histogram of the treelength differences.

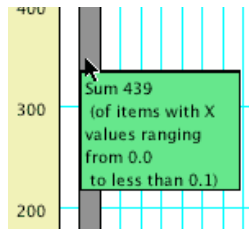
Interpreting the results

An analysis of this exact sort but with 500 replicates yielded the following histogram:



Of the 500 replicates, 439 had a value 0, 58 had a value 1, and 3 had a value 2. You can determine these numbers by touching on each bar with the arrow:





You can also see these values by going to the Text view of the chart (by touching the Text tab at the upper part of the chart window):

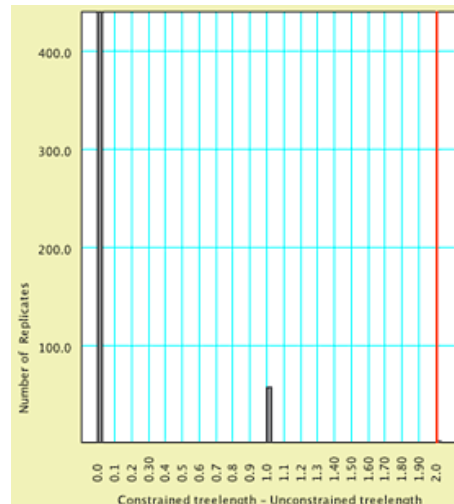
```

▶ / Graphics / Text / Parameters / Modules / Citations
Mesquite version 0.99      Date: Tue Aug 20 19:52:49 MST 2002
=====
Constrained treelength - Unconstrained treelength: Number of Replicates

0: 439.0
1: 58.0
2: 3.0

```

To see what values are in a specified percentile of the left or right tail of the distribution, you can choose (Histogram) Histogram > Analysis > Percentiles.... In the dialog box presented, you can choose the value of the percentile, the color of the bar to be shown, and whether the left or right tails (or both) are calculated. By default, the percentile value is 0.05; on this example, it would be displayed as:



All values to the right of this red bar are thus in the extreme of the distribution, and our simulation would suggest that any value greater than or equal to 2 would occur with a frequency of less than 0.05. Our observation of 9 is thus an unlikely outcome if our hypothesis were true. Thus, we can reject our hypothesis at $p < 0.05$, and conclude that trachypachids and dytiscoids do not form a clade.

As it may be hard to interpret the value of the boundaries by the colored vertical lines, you might want to know the exact values of the percentile boundaries. This information can be gathered by looking at the text view for the histogram. Toward the bottom of the text view in the above example is the following:

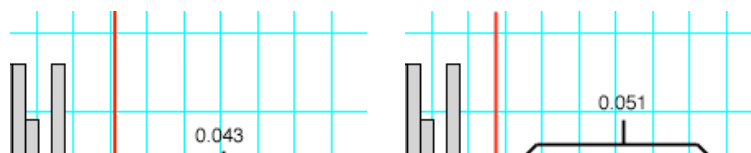
```

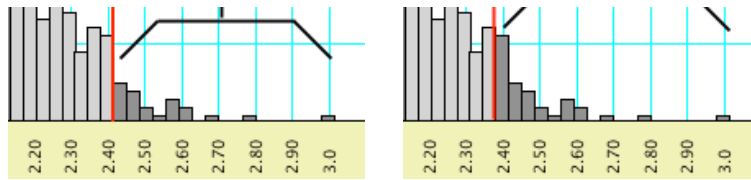
0.0060 percentile boundary (right tail) >= 2.0
0.0060 is the closest percentile to 0.05 that was found,
and corresponds to 3 replicates out of 500.

```

The percentile bar shown will be that that is closest to that requested, but not over it.

The behavior of Mesquite can best be illustrated with another example. If you requested a percentile of 0.05, and if a percentile bar might be placed at 0.043 (left, below), but moving it one increment higher would include enough values to increase the percentile to 0.051 (right, below), then the percentile bar shown will be 0.043.





Summary

The hypothesis tested herein was the monophyly of a clade. The steps in the test are:

- Determine the observed value of the test statistic for the observed DNA sequences. In this case, the treelength of unconstrained most-parsimonious trees was determined using PAUP*, and was then subtracted from constrained (trachypachids with dytiscoids) treelength.
- The best tree matching the hypothesis to be tested (trachypachids with dytiscoids) is inferred using the available sequence data, with branch lengths inferred using maximum likelihood. This is the model tree.
- Values of parameters of a model of sequence evolution (gamma shape parameter, rate matrices, etc.) are inferred using maximum likelihood on the model tree for the DNA sequences.
- The data matrix is opened in Mesquite, along with the model tree. Submodels and a model of character evolution are created within Mesquite to match those inferred.
- Mesquite's Batch Architect is used to automate the process of simulating the evolution of multiple (100 or more) data matrices under this model, using Mesquite's Genesis package. Batch Architect also builds a command file for PAUP* and an instruction file for Mesquite so that it can interpret the results of the PAUP* analyses.
- The command file is executed in PAUP*, telling PAUP* to search for the shortest constrained and unconstrained trees for each of the simulated matrices, accumulating the results into a score file.
- The Mesquite Instructions file is then read into Mesquite, and the score file is read, and Mesquite presents a histogram of the distribution of the test statistic (constrained treelength - unconstrained treelength).
- The observed value of the test statistic is compared to the distribution of the statistic expected under the model as determined by the simulations. If the observed value is more extreme than that expected (say, greater than 95% of the expected values), the hypothesis is rejected.

References

- Bell, R. T. 1966. *Trachypachus* and the origin of Hydradephaga. *Coleopta Bull.* 20:107-112.
- Felsenstein, J. 1978. Cases in which parsimony and compatibility methods will be positively misleading. *Systematic Zoology*, 27, 401-410.
- Goldman, N., J. P. Anderson, and A. G. Rodrigo. 2000. Likelihood-based tests of topologies in phylogenetics. *Syst. Biol.* 49:652-670.
- Hammond, P. M. 1979. Wing-folding mechanisms of beetles with special reference to investigations of Adephagan phylogeny. Pp. 113-180 in *Carabid beetles; their evolution, natural history, and classification* (T. L. Erwin, G. E. Ball, D. R. Whitehead, and A. Halpern, eds.) . W. Junk, The Hague.
- Hendy, M.D. & Penny, D. 1989. A framework for the quantitative study of evolutionary trees. *Systematic Zoology*, 38, 297-309.
- Huelsenbeck, J.P., Hillis, D.M. & Jones, R. 1995. Parametric bootstrapping in molecular phylogenetics: Applications and performance. *Molecular zoology: Advances, Strategies, and Protocols; Symposium held during Annual Meeting of the American Society of Zoologists, St. Louis, Missouri, USA, January 5-8, 1995* (ed. by J. D. Ferraris & S. R. Palumbi), pp. 19-45. Wiley-Liss, Inc., New York.
- Huelsenbeck, J.P. 1997. Is the Felsenstein zone a fly trap? *Systematic Biology*, 46, 69-74.
- Huelsenbeck, J. P., D. M. Hillis, and R. Nielsen. 1996. A likelihood-ratio test of monophyly. *Syst. Biol.* 45:546-558.
- Roughley, R. E. 1981. Trachypachidae and Hydradephaga (Coleoptera), a monophyletic unit? *Pan-Pac. Entomol.* 57:273-285.
- Shull, V., A.P. Vogler, M.D. Baker, D.R. Maddison, and P.M. Hammond. 2001. Sequence alignment of 18S ribosomal RNA and the basal relationships of adephagan beetles: Evidence for monophyly of aquatic families and the placement of Trachypachidae. *Systematic Biology*, 50:945-969.
- Swofford, D. L., G. J. Olson, P. J. Waddell, and D. M. Hillis. 1996. Phylogenetic inference. Pp. 407-514 in *Molecular Systematics* (D. M. Hillis, C. Moritz, and B. K. Mable, eds.) . Sinauer, Sunderland, MA.
- Ward, R. D. 1979. Metathoracic wing structures as phylogenetic indicators in the Adephaga (Coleoptera).

Pp. 181-191 in Carabid beetles; their evolution, natural history, and classification (T. L. Erwin, G. E. Ball, D. R. Whitehead, and A. Halpern, eds.) . W. Junk, The Hague.

Citation for this page

Maddison, D.R. 2003. Testing monophyly of a group of beetles. Study 1 *in* Mesquite: a modular system for evolutionary analysis, version 1.0, <http://mesquiteproject.org>.

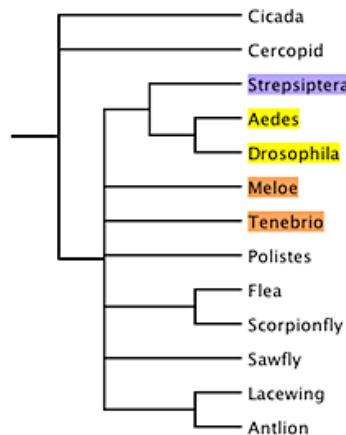
Are strepsipterans related to flies? Exploring long branch attraction

David R. Maddison

The question

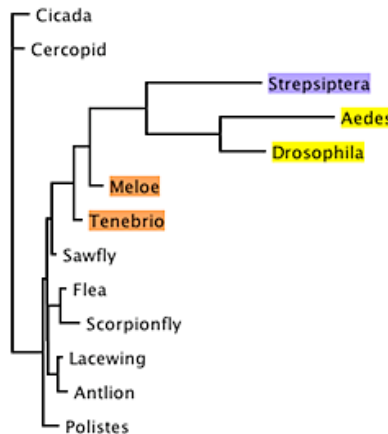
[Strepsiptera](#), sometimes known as twisted-wing parasites, is an enigmatic group of insects. They are parasites of other insects. The males have only one pair of wings and odd raspberry-like eyes and lobed antennae. Females are wingless, and in most species never leave the host. These insects have traditionally been considered related to [beetles](#), although that placement is supported by little evidence. Recently, molecular data have suggested that they may instead be related to true, two-winged flies ([Diptera](#)).

For example, parsimony analysis of a small data matrix of 18S ribosomal DNA yields (Carmean and Crespi, 1995) a phylogeny in which Strepsiptera is placed as sister group of Diptera (in yellow) rather than Coleoptera (in orange):



This is the strict consensus tree of 27 most-parsimonious trees.

However, examining the relative branch lengths for any one of the trees, it becomes evident that the branches for Diptera and Strepsiptera are unusually long:



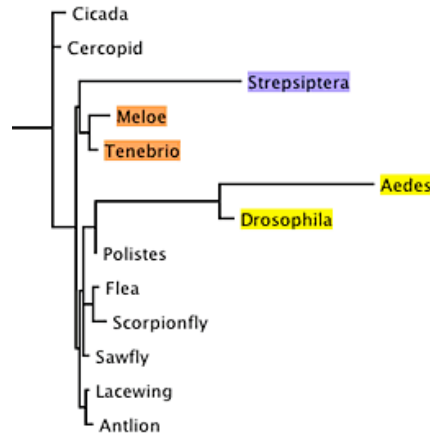
This raised the possibility that the association between Strepsiptera and Diptera in these trees was an artifact, caused by long-branch attraction (Felsenstein, 1978; Hendy and Penny, 1989).

Huelsenbeck (1997) set to determine if the relationship seen in this analysis of Strepsiptera with Diptera could be accounted for by long-branch attraction. To do this, he conducted a simulation study. We won't reproduce his whole study here, just one part of his Figure 2.

We will ask only one simple question: If strepsipterans are indeed related to beetles, would our observation that parsimony analysis yields a phylogeny with strepsipterans related to Diptera be unexpected? If we can show that it is unexpected, we can reject the notion that strepsipterans are related to beetles. If, however, inference of strepsipterans as related to Diptera is expected even if they are actually related to beetles, then we cannot so readily reject the traditional view.

A simulation study

To conduct a simulation study, we first need a model tree. We can use a tree inferred using maximum likelihood, which places strepsipterans with beetles:



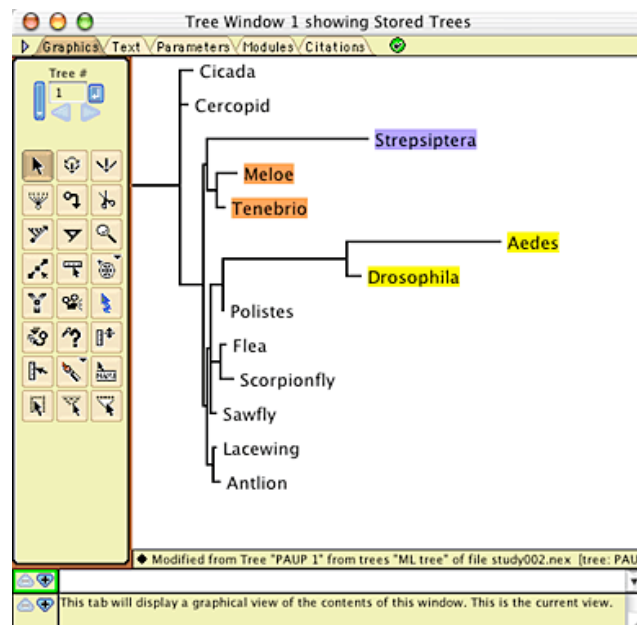
Model Tree

The branch lengths of this tree were inferred using maximum likelihood. We also need a full model of DNA sequence evolution (which can also be inferred using maximum likelihood).

With this model in hand, we can simulated the evolution of 18S rDNA up the branches of this phylogeny, to yield a simulated matrix. We can then infer the phylogeny for this simulated matrix using parsimony, and see where Strepsiptera falls. Repeating this multiple times will give us an idea about the expected placement of Strepsiptera using parsimony inference presuming the model tree shown above.

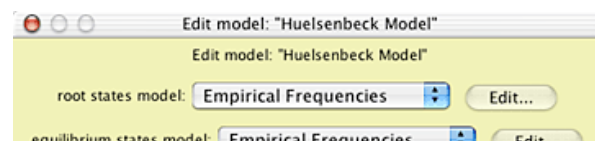
Conducting the simulations

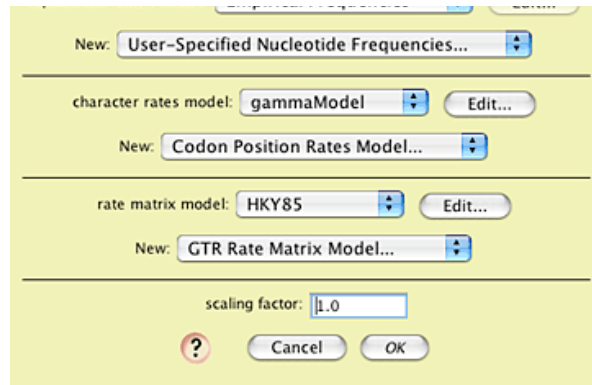
To conduct the simulations, in Mesquite open the example file "study002.nex", in the folder Mesquite_Folder/docs/mesquite/studies/study002/. This file contains the 13-taxon data matrix from Carmean and Crespi (1995) as modified by Huelsenbeck (1977), as well as the model tree shown above. The model tree will appear in a window when you open the file:



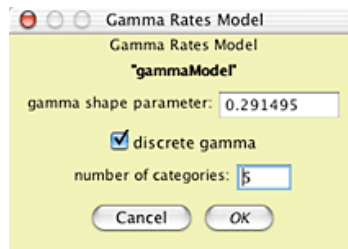
In addition, a model of character evolution has been entered into this file, with parameters obtained from maximum likelihood inference using PAUP*4 (Swofford, 2003). These were inferred on the model tree using the observed 18S rDNA data.

You can see the model by choosing Characters>Edit Character Model>Huelsenbeck Model:

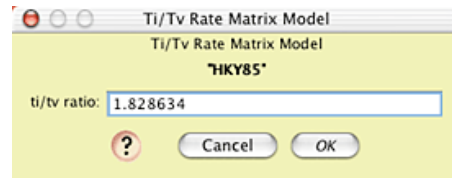




This model uses the empirical state frequencies as present in the original 18S rDNA matrix, a model of character rate variation called "gammaModel", and a rate matrix model called "HKY85". You can see the nature of gammaModel by touching on the Edit button near its listing:



and the nature of the HKY85 model by touching on its Edit button:



With the model fully established, we can now conduct the simulation study. To do this, we want to ask Mesquite to create many simulated matrices, each evolved according to the model. We also want to create a script ("batch file") that will tell a tree-inference program (we will use PAUP*, but another program such as NONA could be used) to find the most parsimonious trees for each of the matrices, and summarize the results. Mesquite's Batch Architect package contains the tools to automate this process.

Choose [Analysis>Batch Architect > Export Matrices & Batch Files...](#) In the first dialog box that appears choose Simulated Matrices on Current Tree, then Evolve DNA Characters, then Huelsenbeck Model. You will be presented with the Export Matrices & Batch Files dialog box, in which you can enter the base name for the matrix files to be created and the number of matrices (100 is a good start).



The batch file template to be used is the one called "Basic PAUP tree search". This template builds a PAUP* command file that will tell PAUP to execute and analyze each matrix in turn, and then, at the end, harvest the results and calculate a majority-rule consensus tree.

After pressing OK, you will be asked for a location to save the 100 matrices. It is recommended that you have an empty folder available into which they can be saved to avoid cluttering up another folder with many files. You will be asked one last question: the number of characters to be evolved in each matrix. As we want the simulation model to be as similar as possible to reality, we will choose to evolve the same number of characters as is present in the observed matrix, which is 770.

Mesquite will now simulate the matrices, and produce a batch file called "paupCommands.nex". This file consists of commands for PAUP*. The start of the file looks something like this:

```
#NEXUS
begin paup;
  set autoclose nowarnreset nowarntrise nowarnsave;

  execute 'StrepSim0.nex';
  hs;
  savetrees file = 'StrepSim0.trees';
  contree / strict= yes majrule=no treefile = 'CBStrepSim' append = yes;

  execute 'StrepSim1.nex';
  hs;
  savetrees file = 'StrepSim1.trees';
  contree / strict= yes majrule=no treefile = 'CBStrepSim' append = yes;

  execute 'StrepSim2.nex';
  hs;
  savetrees file = 'StrepSim2.trees';
  contree / strict= yes treefile = 'CBStrepSim' append = yes;
```

After some initial setup, PAUP* is to execute the first simulated matrix file ("StrepSim0.nex"), do a heuristic search (which, by default, should be for most-parsimonious trees), and then save the results to a tree file. It then saves the strict consensus tree of the most-parsimonious trees to a different tree file. It repeats this process for the next matrix, StrepSim1.nex, and the next one, StrepSim2.nex, and so on, until it gets to StrepSim99.nex, as requested near the end of the file:

```
execute 'StrepSim99.nex';
hs;
savetrees file = 'StrepSim99.trees';
contree / strict= yes majrule=no treefile = 'CBStrepSim' append = yes;

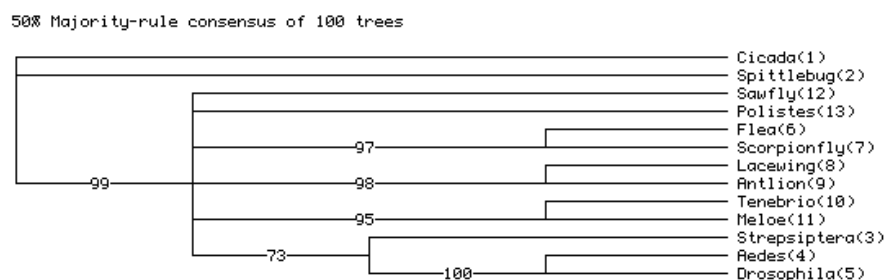
gettrees file = 'CBStrepSim' warntrise = no allblocks;
savetrees file = 'consensus.trees';
contree / strict= yes treefile = 'StrictConsCons.trees';
contree / strict= no majrule=yes treefile = 'MajRuleConsCons.trees';
end;
```

After processing StrepSim99.nex, it reads in the tree file containing the accumulated consensus trees, and calculates both a strict consensus tree of the results from the 100 replicates, as well as the majority rule consensus tree of the replicate's results.

To ask PAUP* to do this analysis, open PAUP* and ask it to execute the file paupCommands.nex.

Interpreting the results

After PAUP* finished executing paupCommands.nex, near the bottom of PAUP*'s main window will be the majority rule consensus tree of the strict consensus trees from each of the 100 replicates. It will look something like this:



Summary

Simulations were done to see what trees we would expect from a phylogeny inference under a particular model tree. The steps in the study are:

- The branch lengths of the model tree are inferred using maximum likelihood in PAUP*.
- Values of parameters of a model of sequence evolution (gamma shape parameter, transition/transversion rate) are inferred using maximum likelihood in PAUP* on the model tree using the observed DNA sequences.
- The data matrix is opened in Mesquite, along with the model tree. Submodels and a model of character evolution are created within Mesquite to match those inferred.
- Mesquite's Batch Architect is used to automate the process of simulating the evolution of multiple (100 or more) data matrices under this model, using Mesquite's Genesis package. Batch Architect also builds a command file for PAUP* and an instruction file for Mesquite so that it can interpret the results of the PAUP* analyses.
- The command file is executed in PAUP*, telling PAUP* to search for the shortest trees for each of the simulated matrices, accumulating the consensus trees for each matrix in a tree file. PAUP* is also instructed to calculate a majority-rule consensus tree of the results of each analysis.
- The majority-rule consensus tree is examined in PAUP* to see what trees are expected to be inferred under these conditions.

References

- Carmean, D., and B. Crespi. 1995. Do long branches attract flies? *Nature*, 373:666.
- Felsenstein, J. 1978. Cases in which parsimony and compatibility methods will be positively misleading. *Systematic Zoology*, 27, 401-410.
- Hendy, M.D., and Penny, D. 1989. A framework for the quantitative study of evolutionary trees. *Systematic Zoology*, 38, 297-309.
- Huelsenbeck, J.P. 1997. Is the Felsenstein zone a fly trap? *Systematic Biology*, 46, 69-74.
- Swofford, D. L. 2003. PAUP*. *Phylogenetic Analysis Using Parsimony (*and Other Methods)*. Version 4 beta 10. Sinauer Associates, Sunderland, Massachusetts.

Citation for this page

Maddison, D.R. 2003. Are strepsipterans related to flies? Exploring long branch attraction. Study 2 in *Mesquite: a modular system for evolutionary analysis*, version 1.0, <http://mesquiteproject.org>.
