

David R. Maddison
University of Arizona

Wayne P. Maddison
University of Arizona



Sinauer Associates, Inc.
Publishers

SUNDERLAND, MASSACHUSETTS, U.S.A.

Copyright 2000 by David R. Maddison and Wayne P. Maddison.
All rights reserved.

To order or for more information contact Sinauer Associates, Inc., Sunderland, Massachusetts 01375 U.S.A, or on the web at <http://sinauer.com>

Names of products and companies mentioned in this manual may be trademarks of their respective companies and are hereby acknowledged. The Quicktime icon is a trademark of Apple, Inc.

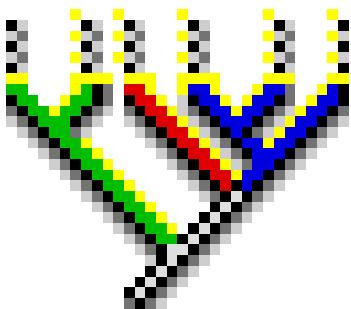


TABLE OF CONTENTS

| | |
|--|---------------------|
| Preface | 9 |
| Preface to Version 3 | 11 |
| 1. Introduction | 13 |
| MacClade..... | 13 |
| Learning how to use MacClade | 17 |
| Conventions | 18 |
| How to cite MacClade | 18 |
| Earlier versions of MacClade | 19 |
| Differences between MacClade 4 and 3 | 19 |
| Future plans for MacClade | 20 |
| Registration..... | 20 |
| Electronic mail and World Wide Web addresses | 20 |
| Reporting bugs | 20 |
| Technical information and availability of source code..... | 21 |
| 2. A Tutorial Overview of MacClade | 22 |
| Tree manipulation | 22 |
| Entering data | 29 |
| Specifying assumptions | 31 |
| Manipulating molecular sequence data | 32 |
| Charting results | 34 |
| 3. Overview of Phylogenetic Inference..... | 37 |
| Trees and their terms..... | 37 |
| Characters and their terms | 42 |
| Inferring the tree | 44 |
| Using phylogenetic trees to interpret evolution | 51 |
| Character evolution: History versus models | 51 |
| Inferring the history of character change | 54 |
| Inferring models of character change using phylogenies | 60 |
| What assumptions are acceptable? | 64 |
| Parsimony versus other criteria..... | 65 |
| 4. Reconstructing Character Evolution Using Parsimony..... | 66 |
| Goals, assumptions, and algorithms..... | 66 |
| Ancestral states and reconstructions..... | 67 |
| Assumptions about character evolution..... | 69 |
| Finding the most parsimonious ancestral states | 76 |
| Equally parsimonious reconstructions | 96 |
| Uncertainty in the reconstruction | 100 |

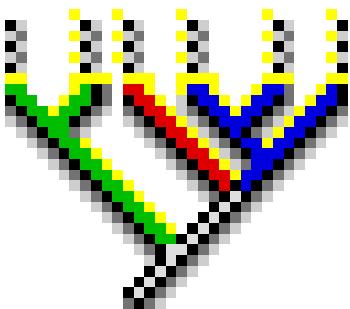
| | |
|---|------------|
| Statistics derived from reconstructions | 101 |
| Locating and summarizing changes of character state | 103 |
| Polytomies and their difficulties | 105 |
| Uncertainty in the phylogenetic tree | 108 |
| 5. Getting Started with MacClade | 110 |
| Hardware and system software specifications | 110 |
| Installing MacClade | 110 |
| Using MacClade under Microsoft Windows® | 111 |
| Updates to MacClade | 111 |
| Full and Classroom versions of MacClade | 111 |
| Computer memory and MacClade's limits | 112 |
| Starting MacClade | 113 |
| Getting help | 114 |
| Messages | 115 |
| Setting preferences for MacClade's options | 115 |
| Using MacClade with other programs | 117 |
| Quitting from MacClade | 118 |
| 6. Managing Data Files | 119 |
| Creating a new MacClade data file | 119 |
| Opening an existing data file | 119 |
| Saving files to disk | 121 |
| The format of MacClade files | 121 |
| Saving files in other formats | 122 |
| Altering the way MacClade writes files | 122 |
| Reverting to last saved version | 123 |
| Closing data files | 123 |
| Editing restrictions in the classroom version | 123 |
| 7. NEXUS Files and Blocks | 125 |
| The NEXUS format | 125 |
| Processed and foreign NEXUS blocks | 126 |
| Cursors displayed while reading NEXUS files | 127 |
| Changing the way MacClade writes processed NEXUS blocks | 128 |
| NEXUS block management | 130 |
| Preparing a NEXUS format file by hand | 137 |
| Changes in the NEXUS format | 138 |
| 8. Importing and Exporting Text Files | 139 |
| Importing files | 139 |
| Exporting files | 140 |
| MacClade 1.0 and 2.1 files | 141 |
| MacClade 3 files | 141 |
| PAUP* files | 141 |
| PHYLIP files | 141 |
| HENNIG86 files | 143 |
| NONA files | 144 |
| NBRF files | 144 |
| CLUSTAL, GCG/MSF, FASTA, GenBank, and GenPept files | 145 |
| Simple table files | 145 |

| | |
|---|------------|
| Taxon name lists | 147 |
| Importing sequences into an existing file | 147 |
| Exporting descriptions | 150 |
| Exporting MEGA files | 151 |
| Exporting Web pages (HTML files) | 151 |
| 9. MacClade's Windows | 155 |
| The Windows menu | 155 |
| Closing windows | 155 |
| Data editor | 155 |
| State names & symbols window | 156 |
| List windows | 157 |
| Tree window | 157 |
| Tool palettes | 158 |
| Chart window | 159 |
| Text windows for keeping notes | 160 |
| 10. List Windows | 161 |
| The list windows | 161 |
| Structure of a list window | 162 |
| Selecting objects | 163 |
| Setting values for selected characters and taxa | 175 |
| Using or displaying single selected objects | 175 |
| Rearranging the order of objects | 175 |
| Deleting objects | 178 |
| Naming objects | 179 |
| 11. Taxa | 180 |
| Creating new taxa | 180 |
| Importing taxa or sequences | 180 |
| Duplicating existing taxa | 180 |
| Creating a consensus sequence taxon | 181 |
| Naming taxa | 181 |
| Reordering taxa | 185 |
| Deleting taxa from the matrix | 186 |
| Excluding and including taxa from the tree | 186 |
| Merging taxa | 186 |
| Filtering redundant taxa | 187 |
| Adding notes and pictures about taxa | 188 |
| Defining your own taxon sets | 189 |
| 12. Characters and Their States | 192 |
| Data formats | 192 |
| Naming characters | 193 |
| State names and symbols | 194 |
| Symbols for missing data, gaps, AND and OR | 199 |
| Creating new characters | 200 |
| Duplicating existing characters | 200 |
| Reordering characters | 200 |
| Compressing characters | 200 |
| Deleting characters from the matrix | 201 |
| Excluding characters from analyses | 201 |

| | |
|--|------------|
| Specifying assumptions about characters | 201 |
| Adding notes and pictures about characters | 201 |
| Defining your own character sets | 201 |
| 13. Entering and Editing Data..... | 204 |
| Creating and editing a data matrix | 204 |
| Getting to the data editor | 206 |
| Moving around the matrix | 206 |
| Data editor tools | 207 |
| Selecting elements in the editor | 210 |
| Undoing changes in the editor | 218 |
| Adding taxa or characters | 219 |
| Deleting taxa, characters, and cell blocks | 222 |
| Reordering taxa or characters | 223 |
| Entering data | 224 |
| Recoding character data | 237 |
| Moving data cells & sequence alignment | 240 |
| Finding sequences of states | 252 |
| Proofreading your matrix | 254 |
| 14. Display of Data | 255 |
| Display of data cells | 255 |
| Coloring characters | 266 |
| Other display options | 269 |
| 15. Assumptions About Characters | 273 |
| Displaying and changing current assumptions | 273 |
| Character weights | 274 |
| Character exclusion | 278 |
| Protein-coding regions | 281 |
| Character transformation assumptions | 282 |
| 16. Preparing Molecular Data | 292 |
| Importing data from molecular databases | 292 |
| Working with alignment programs | 292 |
| DNA, RNA, and protein data formats | 292 |
| Specifying coding regions and codon positions | 292 |
| Consensus sequences | 295 |
| Alignment of molecular sequences within MacClade | 297 |
| Aligning nucleotides to match a protein alignment | 299 |
| Other data editor features for sequence data | 301 |
| Setting the genetic code | 305 |
| Translating nucleotide sequences to amino acid sequences | 307 |
| Creating protein-parsimony user-defined types | 309 |
| File saving options | 309 |
| 17. Trees and Tree Manipulation | 310 |
| Trees in MacClade | 310 |
| MacClade's tree window | 311 |
| Getting to the tree window | 311 |
| Tree window structure and size | 312 |
| Tree manipulation: Changing cladistic structure | 317 |

| | |
|--|----------------------------|
| Tree display commands | 334 |
| Tree storage and retrieval | 340 |
| 18. Tracing Character Evolution. | 343 |
| Tracing the history of character evolution | 343 |
| Interpreting the character tracing | 345 |
| Manipulating ancestral states..... | 350 |
| Resolving equivocal tracings | 353 |
| Display of character tracing | 355 |
| Character Data Boxes | 357 |
| Summary of all changes on branches: Trace All Changes..... | 358 |
| Branch lengths | 370 |
| Summary of all states along each branch | 371 |
| Confirming the validity of the parsimony algorithms | 371 |
| 19. Basic Tree and Character Statistics | 373 |
| Display of statistics | 373 |
| Number of steps | 374 |
| Treelength | 374 |
| Number of changes | 375 |
| Tree changes | 375 |
| Minimum and maximum possible steps | 376 |
| Consistency and retention indices | 377 |
| Calculating the percentage of missing data and gaps | 379 |
| Calculating nucleotide percentages for each sequence | 380 |
| Calculating the number of stop codons in protein-coding nucleotide sequences | 381 |
| 20. Charting Tree and Character Statistics | 382 |
| General issues concerning chart calculations | 383 |
| Character Steps/etc. | 389 |
| Treelengths of many trees..... | 394 |
| State Changes & Stasis | 396 |
| States | 404 |
| Comparing two trees | 404 |
| Comparing two tree files | 405 |
| Chart display | 406 |
| Examples using molecular data | 411 |
| Confirming the validity of the charting calculations | 417 |
| 21. Continuous Characters. | 418 |
| To make discrete or not?..... | 418 |
| Editing continuous characters | 418 |
| Tracing continuous characters | 420 |
| Confirming the validity of the parsimony algorithms | 422 |
| 22. Patterns of Correlated Character Evolution. | 423 |
| Using reconstructed histories to test correlation..... | 423 |
| The concentrated-changes test | 423 |
| Confirming the validity of the correlation test algorithms | 431 |
| 23. Generating Random Data and Random Trees | 432 |
| Random number generator and its seed | 432 |

| | |
|---|------------|
| Random data | 433 |
| Random trees | 439 |
| 24. Note Keeping | 442 |
| Adding notes about the data file | 442 |
| Adding notes about the trees | 442 |
| Annotating cells in the data matrix | 443 |
| 25. Recording Your Work: Printing, Graphic, and Text Files | 448 |
| Saving a record of your work | 448 |
| Printing the data matrix | 448 |
| Printing character and state names | 453 |
| Printing list windows | 453 |
| Printing text windows | 453 |
| Printing trees | 453 |
| Printing charts | 464 |
| Printing other components | 464 |
| Saving graphics files | 466 |
| Saving text files | 468 |
| Placing a text version of the tree into the Clipboard | 470 |
| 26. Using MacClade and PAUP* Together | 471 |
| A scenario of using MacClade and PAUP* | 471 |
| Using both programs simultaneously | 471 |
| Control over NEXUS blocks | 472 |
| Embedding PAUP* commands in data files | 472 |
| Calculating decay index values | 472 |
| Incompatibilities between MacClade and PAUP* | 474 |
| Differences in data file format | 479 |
| 27. References | 481 |



PREFACE

In the years since MacClade 3's initial 1992 release, phylogenetic biology has undergone some notable changes. Phylogenetic thought has so thoroughly penetrated evolutionary biology, including population genetics and evolutionary ecology, that phylogenetic trees have invaded many previously ahistorical journals. There has been a proliferation of new or refined methods of analysis, including maximum likelihood and others, whose popularity has benefited from the availability of molecular sequence data and the increasing speed of computers.

We anticipate that MacClade 4's role in today's phylogenetic biology will continue that of MacClade 3: exploring phylogenetic trees and character data, editing data with its specialized spreadsheet editor, and interpreting character evolution using parsimony. In tune with the abundance of molecular data, the data editor has many enhancements for handling DNA and protein sequences, including sequence alignment and new visualization tools. Other enhancements include a uniform interface (list windows) for managing characters, taxa, trees and other items, and some additional features in the tree window. There are separate versions for older Macintoshes that use 68K processors and newer ones using PowerPC processors. For those of you who have used version 3, [Chapter 1](#) gives a brief overview of some important differences between version 3 and version 4.

MacClade 3 was built to coexist with other programs, sharing files and performing complementary functions. With the growing number of programs available, sharing and complementarity are even more important now. MacClade 4 continues this theme, with enhanced facilities to import and export files, and new methods to interact with Swofford's PAUP* program. We have been planning for coexistence as well with another effort of ours: Mesquite (see [page 20](#)). While MacClade 4 was being developed, we explored the possibility of adding to it many additional phylogenetic calculations, but realized that it would be technically difficult and would compromise the compactness and tight integration of its existing features. And so we split our efforts into two: MacClade as a compact, easy-to-use program superior in sequence editing and parsimony calculations, and Mesquite as a modular system with functions ranging through population genetics, morphometrics, and molecular evolution.

With version 4 of MacClade the order of authorship has changed. For version 3, our decision regarding the order of authorship had been a difficult one, being a balance of origination and some key contributions (Wayne had conceived and developed MacClade originally, and was responsible for the core parsimony calculations and most of chapters 3–5 in the 1992 book) versus total effort (David had been responsible for more than two-thirds of the programming code for version 3, including many of the new features). With version 4 the decision has been easy: David is responsible for almost all of the enhancements since version 3, including greatly improved parsimony calculations for implicitly examining all MPRs, and in total is responsible for more than 80% of the lines of programming code in version 4. For those curious about the history of MacClade, we have included on this CD-ROM the manuals accompanying versions 1.0, 2.1, and 3.

Many people aided in the development of MacClade 4, in addition to those mentioned in ["Preface to Version 3"](#). We thank our beta testers, who aided us by finding numerous entomological gems, and by giving us excellent suggestions for improvement. Derek Sikes, Christoffer Schander, David Fitch, Gonzalo Giribet, Sean Graham, Betsy Arnold, Kipling Will, Dave Swofford, Patrik Lindenfors, Elizabeth Jockush, Buz Wilson, Marshal Hedin, Karen Ober, Volker Gurtler, Larry Buckley, Tim Collins, Andrew Smith, Mary Liz

Jameson, Joana Carneiro da Silva, Michel Laurin, Joel Dacks, Lucinda McDade, Rachel Levin, Vincent Savolainen, David Nickle, Chuck Delwiche, Aniko Sabo, Bill Birk, Patrick Abbot, John Stireman, Michelle Zjhra, Cliff Cunningham, David Reed, Carlos Lopez Vaamonde, Melissa Fleming, Karl Kjer, Harry Erwin, Gustavo Ybazeta, David Posada, Gavin Naylor, Elizabeth Sinclair, Richard H. Thomas, Andrew McArthur, and David Fitch, among others, discovered various "suboptimalities" in MacClade or made suggestions. Of special note are Margaret Thayer and Bill Piel, whose thoroughness at finding and reporting bugs is very much appreciated. Des Higgins provided suggestions that improved the pairwise alignment tool. Thanks also go to Betsy Arnold and Bobbie Lewis for reviewing the manual in detail.

In our efforts to get MacClade running on the Microsoft Windows operating system using the MacOS emulator Executor, Cliff Matthews, the author of Executor, went out of his way to help, and thereby not only improved MacClade's performance under Executor, but on the MacOS as well.

For their patience and support, we thank our three children, Julia, Christopher, and Teresa, as well as Helen, Betsy, and Leticia.

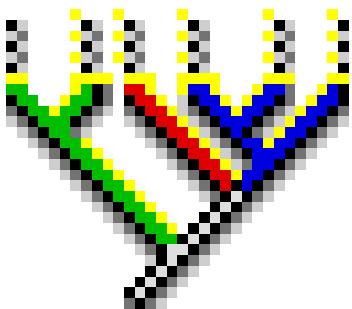
As always, it has been a pleasure to work with Andy Sinauer and his staff, including Kathleen Emerson, and we thank them all for their encouragement, effort, and attention to quality.

David R. Maddison

Department of Entomology
University of Arizona
Tucson, AZ 85721

Wayne P. Maddison

Department of Ecology
and Evolutionary Biology
University of Arizona
Tucson, AZ 85721



PREFACE TO VERSION 3

MacClade began in 1985 when one of us (WPM) began to explore drawing and interacting with trees on the screen of an Apple Macintosh™ computer. From these initial explorations sprang MacClade version 1.0, which was released in June of 1986 (W. Maddison, 1986). It contained the central features of the Tree Window — namely, tracing of characters with fixing of the states at branches, rearrangement of branches of the tree using the computer's mouse, and a feature like Trace All Changes. It was primitive in other features.

MacClade version 2.1 was released in March of 1987 (Maddison and Maddison, 1987). It added the full Macintosh user interface of menus and dialog boxes, incorporated a built-in text editor, and added user-defined matrices of state-to-state distances ("step matrices").

Version 3 incorporates many new features beyond those found in version 2.1; some of the most noticeable are the specialized data editor, the chart window, acceptance of polytomous trees, and support for molecular data. For those of you who have used version 2.1, Chapter 1 gives a brief overview of some important differences between version 2.1 and version 3.

Following the original release of version 3.0 in 1992, several minor upgrades have been released. As of this writing, the current version is 3.08. The program is little changed since version 3.0, primarily differing in the addition of a few new features (for example, the ability to draw branch lengths proportional to reconstructed number of changes in Trace All Changes mode, an increase to 10 in the number of continuous characters allowed, and new random data generation options) and in bug fixes. This book differs from that published in 1992 only in the documentation of those few new features and in several corrections of typographical and similar errors.

Many people aided in the development of MacClade. We would like to thank H. W. Levi and E. O. Wilson, who gave us support and encouragement, and the freedom to work on MacClade when we should have been working on our theses. Some of the recent work on MacClade was done while WPM was a Natural Sciences and Engineering Research Council of Canada Post-doctoral Fellow, in Monty Slatkin's lab, and while DRM was a Alfred P. Sloan Foundation Post-doctoral Fellow, in R. C. Lewontin's lab; we thank them for their support.

We thank all those who reviewed pre-release copies of MacClade for us, and tested its functioning. Foremost among these was Willem Ellis, who searched long and hard for bugs in the program, and provided us with many thoughtful reports on MacClade. David Swofford, David Stern, Jeremy Ahouse, Darrel Frost, Ray Phillips, Robert O'Hara, Jenny Chappill, and Todd Disotell also deserve special note for their efforts in keeping us informed about MacClade bugs. Others who found bugs or gave suggestions include Belinda Chang, Jonathan Coddington, Chuck Crumly, Mike Cummings, Michael Donoghue, Jim Doyle, Sharon Gowan, Mike Hammer, Toby Kellogg, Jean-François Landry, Jim Malusa, Santiago Madriñan, Ted Oakes, Norm Platnick, and Jim Woolley. We are grateful for the encouragement we received from those testing the program, especially Michael Donoghue, David Swofford, Willem Ellis, and Mike Cummings. We thank the many users of MacClade 3.0–3.07 who sent us reports of difficulties and helped us to track down problems.

We would like to thank specifically several people who had an impact on the content of MacClade: David Swofford, who was always there for lively discussions about phylogenetics programs, and who provided

programming advice, suggestions for improvement, and code for the random number generator; J. S. Ashe, whose interest in the charts summarizing statistics over multiple trees convinced us that it was worth the effort to include; Scott Edwards, whose desire to see the distribution of transitions and transversion over the length of a DNA sequence inspired the inclusion of Restrict Changes in the Character Steps charts; Allan Wilson and Linda Vigilant for inventing circular trees.

The book was improved by the inclusion of Dan Fisher's chapter on stratocladistics; we especially thank him for being willing to publish an outline of his ideas here. For comments, suggestions and discussion that contributed to this book, we thank Helen Amerongen, Leticia Avilés, David Baum, Michael Donoghue, Willem Ellis, Joe Felsenstein, Daniel Fisher, Larry Gall, Junhyong Kim, Michael Sanderson, and David Swofford.

We are very grateful to Michael Donoghue, Walter Fitch, Arnold Kluge, James Lake, Diana Lipscomb, and Mark Stoneking for supplying electronic versions of data files which form the basis of some of the example data files distributed with MacClade.

MacClade would not have arisen if it were not for Apple Computer's development of the computer and graphical interface that made it possible. We also thank them for supporting our work with a generous award.

Throughout the long gestation period of this current version of MacClade, Andy Sinauer has been incredibly patient and tolerant of the many delays. We also thank Joe Vesely, Carol Wigg, Roberta Lewis, Kathleen Emerson, and Dean Scudder for their efforts.

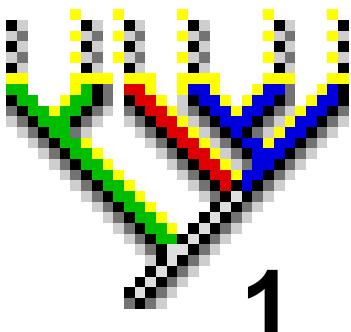
Finally, we would like to thank Leticia Avilés and Helen Amerongen, for their support through many long hours of programming, and for their endurance of horrendous phone bills.

Wayne P. Maddison

Department of Ecology
and Evolutionary Biology
University of Arizona
Tucson, AZ 85721

David R. Maddison

Department of Entomology
University of Arizona
Tucson, AZ 85721



INTRODUCTION

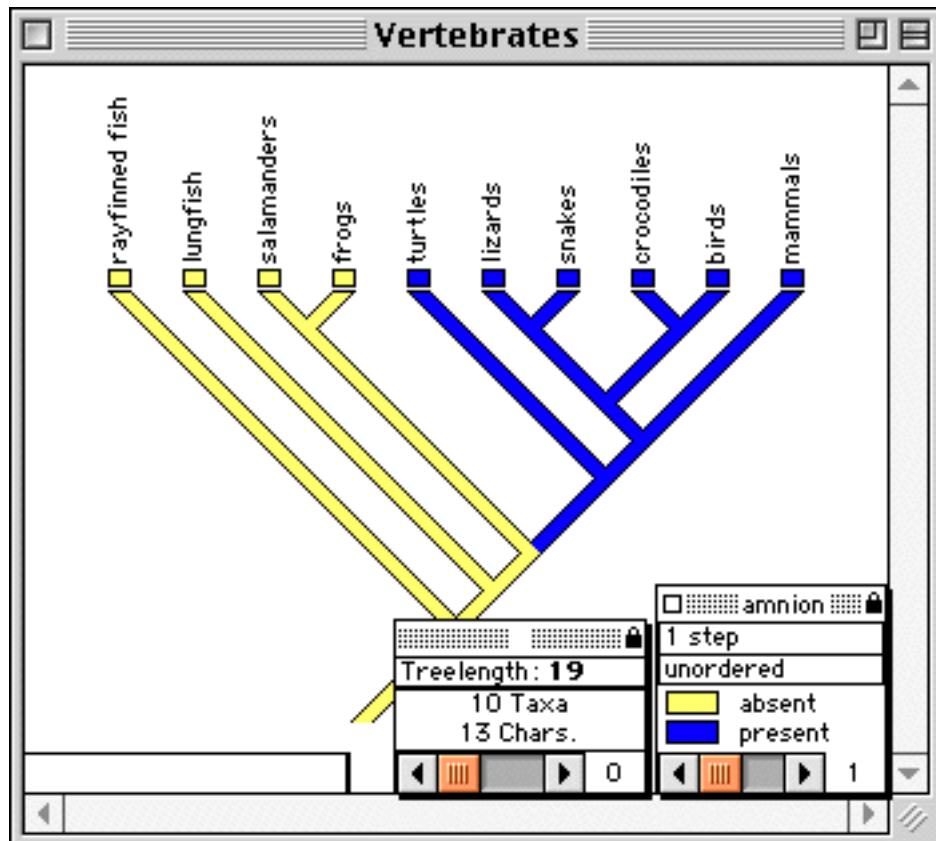
1

MacClade

MacClade is used as a tool for phylogenetic analysis. But by its nature it also embodies a world view, a portrayal of a phylogenetic approach to studying diversity and evolution. It is relatively easy to see the diversity of living organisms, but it has proved more difficult to see that diversity in terms of its history; the slow development of a thoroughly phylogenetic perspective in biology attests to this challenge. Together this manual and program present methods for analyzing and exploring phylogenetic hypotheses, including hypotheses about character evolution. MacClade is one attempt to give our mind's eye phylogenetic lenses, to help us think about and see lineages and evolution.

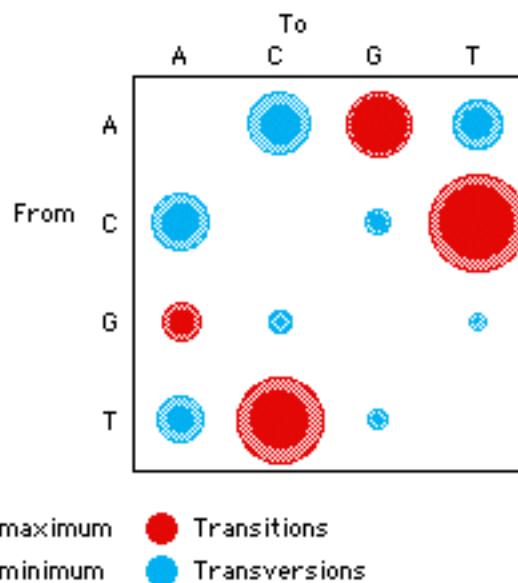
Some of the chapters and sections of this manual (especially [Chapter 3](#) and [Chapter 4](#)) discuss phylogenetic principles in general. Much of the rest of the manual is devoted to MacClade itself. In this introductory chapter, we give a brief overview of MacClade's capabilities, followed by a summary of the chapters in this manual, as well as information on citing MacClade, earlier versions of the program, future versions of MacClade, reporting bugs, and acquiring technical data about the program.

MacClade provides an interactive environment for exploring phylogeny. In MacClade's tree window, hypothesized phylogenetic trees or cladograms can be manipulated and character evolution visualized upon them. To manipulate the tree, tools are provided to move branches, reroot clades, create polytomies, and search automatically for more parsimonious trees. Character evolution is reconstructed on the tree and indicated by "painting" the branches. Alternative reconstructions of character evolution can be explored. Summaries of changes in all characters can be depicted on the tree. As trees are manipulated,



MacClade updates statistics such as treelength and the results are depicted in graphics and charts.

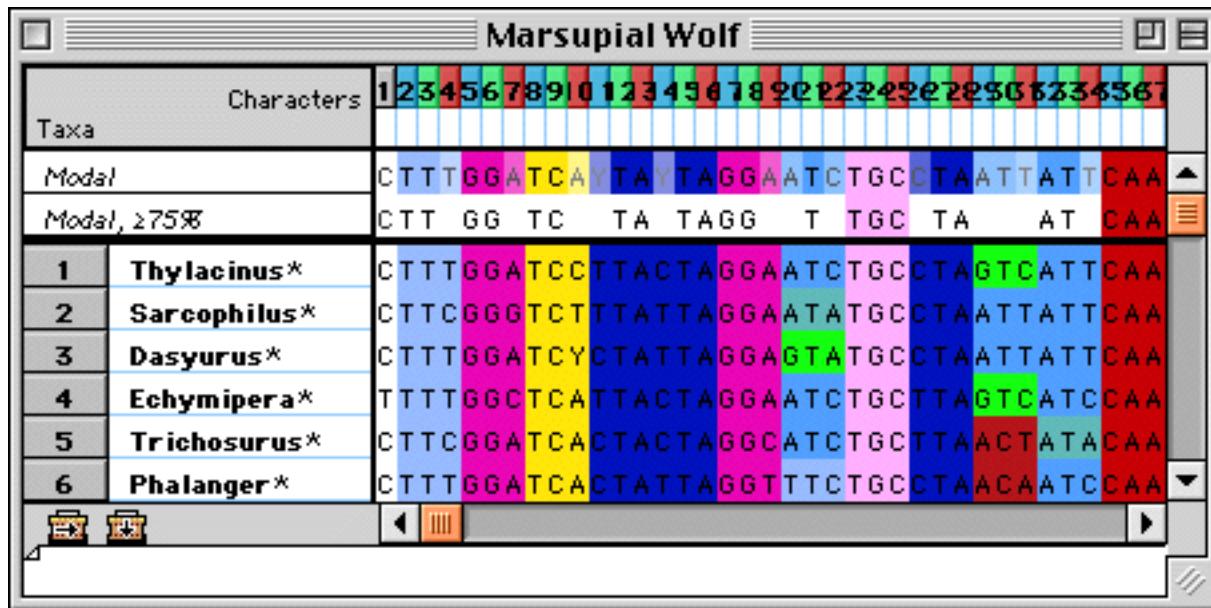
MacClade provides charts summarizing various aspects of character evolution on one or more trees, as well as charts comparing two or more trees. For example, the charts can show the number of trees of each length, the number of characters on the tree with different consistency indices, and so on. There are charts specifically designed for DNA/RNA sequence data, including one showing the number of changes on the tree at first, second, and third codon positions, and a chart of the relative frequencies of various transitions and transversions.



In MacClade's data editor, systematic and comparative data are entered and edited. The editor has numerous features for manipulating rows, columns, and blocks of data, and for recoding data.

| | | Vertebrates | | | |
|------|----------------|-------------|------------|---------------|--------------|
| | | 1 | 2 | 3 | 4 |
| Taxa | Characters | amnion | appendages | body covering | thermoreg. |
| 1 | rayfinned fish | absent | fins | derm.scales | poikilotherm |
| 2 | frogs | absent | legs only | smooth | poikilotherm |
| 3 | turtles | present | legs only | epid.scales | poikilotherm |
| 4 | lungfish | absent | fins | derm.scales | poikilotherm |
| 5 | salamanders | absent | legs only | smooth | poikilotherm |
| 6 | crocodiles | present | legs only | epid.scales | poikilotherm |
| 7 | lizards | present | legs only | epid.scales | poikilotherm |
| 8 | birds | present | legs+wings | feathers | homeotherm |

Many display features and tools in the editor were designed specifically for graphical manipulation and alignment of molecular sequences.



Windows listing characters, taxa, trees, taxon sets, character sets, and so on provide a uniform interface for editing their properties, sorting, and selecting them.



If we had to admit a grand purpose to the features of MacClade, it might be "to help biologists explore the relationships between data and hypotheses in phylogenetic biology". We envisage MacClade's use by biologists of many backgrounds. For example, suppose:

1. A systematist is working on the phylogenetic relationships of snail species. She enters data for 100 morphological characters and 20 taxa in MacClade's data editor, saves the file, and reads it into Swofford's (2000) PAUP*. PAUP* is used to find parsimonious trees, but to her surprise, the resulting trees separate two species she had thought closely related. By moving back to MacClade, examining alternative trees by hand, and using the charting functions, she discovers that the unexpected result is due to the influence of two characters of the nervous system. This not only provokes her to examine those characters in more detail, but also to gather molecular data. The molecular data set, on 60 taxa and 1500 nucleotides, is too large to get exact solutions from PAUP*. PAUP*'s heuristic algorithms, combined with her ability to suggest and examine alternative trees in MacClade, convince her that the optimal trees for both data sets agree that the two species are not sister species. Although she had no intention of looking at fossil species, she realizes that this result suggests that their peculiar shared morphology might be primitive and old. By using the stratigraphic character type in MacClade, she discovers the minimum age of the ancestor in which the morphology was apparently derived was Paleocene, much older than the group of predators against which the morphology was thought to be a defense.
2. A population biologist has been studying the adaptive advantages of larval dispersal strategies. Although his past studies have focused on measuring risks and fecundities at two study sites in the Gulf of California, he realizes that his hypotheses could be tested by seeing if it accurately predicts a species' strategy according to its ecological position. After talking to a phylogenetic biologist, he reluctantly admits that his question is actually one of phylogenetic correlation between ecology and strategy. He finds a collaborator working on the phylogeny of the group, and together they use MacClade to map the evolution of larval dispersal strategies on the phylogenetic tree. He discovers that the correlation between strategy and ecology is not nearly as strong as he would have hoped, but he notices on MacClade's character tracing that the two groups on the phylogenetic tree with a special dispersal strategy also have the most species. His study shifts to an examination of the influence of dispersal strategy on speciation and extinction. After accumulating known phylogenies from numerous groups and exploring them with MacClade, a clear correlation emerges between speciation success and the special dispersal strategy. Itching to get back to his study sites, he discovers that his newly acquired familiarity with trees can be applied to his populations to examine patterns of dispersal and gene flow using reconstructed gene trees. He uses MacClade's random tree and random character generation to formulate null models to test his population hypotheses.
3. A molecular biologist studying cell surface proteins in a herbivorous insect suspects that a particular domain might be involved in binding secondary compounds made by the host plants. A direct chemical approach to demonstrate binding proves difficult, but the genes coding for this protein have been sequenced in several dozen related species. Using MacClade to help reconstruct the phylogeny of the species and to chart how evolutionary changes of amino acids are distributed along the length of the protein, she finds that this domain is indeed the most rapidly evolving. Furthermore, by comparing the phylogenies of the insect and the plant host using Page's COMPONENT (1993), she makes a convincing case that one of the insect groups arose with a shift to a new plant host that had powerful and diverse secondary compounds. Not only did MacClade reconstruct many amino acid changes along that branch of the insect phylogenetic tree, but it also showed that changes in other amino acids were concentrated in the clade of insects living on noxious hosts. With such evidence that the domain was evolving in response to host secondary compounds, the molecular biologist faced with renewed vigor the direct chemical approach to examining the binding.

4. An evolutionary biologist teaching a basic evolution course wants to introduce students to phylogenetic reconstruction. In a live demonstration MacClade is used first to help students picture phylogenetic trees in their minds' eyes. Then, the concept of fit between a tree and a character is shown via MacClade's character tracing. Students are given an example data set and let loose. Soon they are moving branches around with increasing rapidity, treating the search for parsimonious trees like a video game, competing against one another for shorter trees. In the process they discover that grouping by derived similarity will gain them shorter trees. One of them notices that one character disagrees with most of the remaining characters on whatever tree they can find. Indeed, on the parsimonious trees this character shows many cases of convergence. Another character is found that is similarly convergent, and students discuss what evolutionary processes might lead to the two characters having correlated evolutionary changes.

These scenarios illustrate the varied uses of a phylogenetic perspective, and of MacClade in making it accessible.

Learning how to use MacClade

MacClade is a large and complex program. If you are just getting started with MacClade, we advise you to follow through the examples in [Chapter 2, "A Tutorial Overview of MacClade"](#). This will guide you step by step through simple data files, and thereby introduce you to some of the basic functions of MacClade.

Chapters 3 and 4 concern phylogenetic theory. [Chapter 3, "Overview of Phylogenetic Inference"](#), defines terms of relevance to MacClade, discusses methods of inference of phylogenies, and of patterns and processes of character evolution. [Chapter 4, "Reconstructing Character Evolution Using Parsimony"](#), describes the assumptions and methods MacClade uses in reconstructing character evolution. We recommend that you read the relevant sections of these chapters before using MacClade to study character evolution.

Chapters 5 through 25 of this manual provide more detailed guidance about using MacClade. Although we hope that many of MacClade's features will be obvious from the program itself, some important ones will not. So that you may discover these features, and thoroughly understand MacClade's analyses, we strongly recommend that you at least glance through Chapters 5 through 25.

Chapters 5 through 10 introduce some basics of using MacClade. ["Getting Started with MacClade"](#) gives an overview of hardware and software requirements, installation, memory management, and setting preferences. ["Managing Data Files"](#) tells how to create, open, and save data files. ["NEXUS Files and Blocks"](#) describes how to select options for saving NEXUS files, and how to manage NEXUS blocks. ["Importing and Exporting Text Files"](#) describes how to share files with Clustal, NONA, PHYLIP, and other programs, as well as import and export files in NBRF, Genbank, GCG/MSF, and other formats. ["MacClade's Windows"](#) introduces the windows of MacClade, including the data editor window, the tree window, and the list windows. ["List Windows"](#) explains how items like taxa, characters, and trees can be manipulated using the list windows.

Chapter 11, ["Taxa"](#), describes how observed taxa are created and manipulated. Taxa are the basic units with which characters are associated (via data matrices) and whose relationships are shown by trees.

Chapters 12 through 16 give detailed information about manipulating character data and assumptions. ["Characters and Their States"](#) describes MacClade's data formats and how characters and their states can be named and manipulated. ["Entering and Editing Data"](#) explains the use of MacClade's spreadsheet data editor. ["Display of Data"](#) tells how the data editor's display can be adjusted. ["Assumptions About Characters"](#) details the various assumptions used in MacClade's calculations, such as weighting of characters and how states transform one to another. ["Preparing Molecular Data"](#) focuses on data formats and other issues related to molecular data, including sequence alignment.

Chapters 17 through 23 describe working with phylogenetic trees and the relationship between character data and the tree. "[Trees and Tree Manipulation](#)" introduces MacClade's tree window, and how trees can be manipulated, stored, and retrieved. In "[Tracing Character Evolution](#)", methods for reconstructing the ancestral states (and thereby pathways of character evolution) are discussed. In "[Basic Tree and Character Statistics](#)", simple statistics of treelength and indices such as the consistency index are outlined. MacClade's charting facility is described in "[Charting Tree and Character Statistics](#)". Use of continuous-valued characters is explained in "[Continuous Characters](#)". The next chapter discusses "[Patterns of Correlated Character Evolution](#)", particularly the concentrated-changes test for phylogenetic character correlation. MacClade's randomization facilities, which can be used in various statistical tests, are discussed in "[Generating Random Data and Random Trees](#)".

Chapter 24, "[Note Keeping](#)" explains how to attach notes and pictures to various parts of the data file. Chapter 25, "[Recording Your Work: Printing, Graphic, and Text Files](#)" gives strategies for keeping records. Printing facilities are described there.

The remaining chapter ("[Using MacClade and PAUP* Together](#)") describes how to use MacClade in combination with Swofford's PAUP* program (Swofford, 2000).

Conventions

Throughout this manual, menu items and dialog boxes will be shown in **boldface** type. Buttons in dialog boxes will be shown in the font Helvetica. Hypertext is shown blue and underlined, indicating that touching on this text will take you to the described destination.

Important comments will be flagged with labels, as follows:

WARNING: *Important warnings are indicated in this fashion.*

NOTE: *Notes, shortcuts or tips are indicated in this fashion.*

EXAMPLE: *Exercises using the example files included with MacClade are indicated in this fashion.*



For some examples, a MacClade icon appears in the margin. If you touch on this icon, the data file discussed will be opened in MacClade. This feature may not function if you are examining this manual on a MacOS emulator, or the Examples folder is no longer in the same folder as this manual.



QuickTime® videos of some procedures are included with this documentation. Touch on the QuickTime icon in the margin to display the relevant video. This feature will not function if the Movies folder is no longer in the same folder as this manual.

How to cite MacClade

The computer program MacClade is not just a calculating tool. As with a published paper, it contains new ideas, implicit suggestions, and new methods. MacClade is distinct from works published on paper in that new ideas and methods presented are in a form that can be immediately used by researchers and students; it responds actively to a user instead of passively to a reader. Use of ideas in this manual or in MacClade must be acknowledged as with any other published paper. (This may seem obvious; we mention it only as history suggests that it is not.) MacClade should also be cited in your Materials and Methods as a source of the calculations or output used in your papers.

Despite the scientific content of MacClade and other phylogenetic programs such as PAUP* (Swofford, 2000), HENNIG86 (Farris, 1988), and PHYLIP (Felsenstein, 1993), some journal editors may object to placing a citation to a computer program in the Literature Cited. However, with the 21st century here, we think this attitude will become rarer. We recommend the following citation, which can be used both for the program and this accompanying manual:

Maddison, D. R. and W. P. Maddison, 2000. MacClade 4: Analysis of phylogeny and character evolution. Version 4.0. Sinauer Associates, Sunderland, Massachusetts.

Earlier versions of MacClade

Previous versions of MacClade are numbered 1.0, 1.01, 2.1, 3.0, 3.01, 3.02, 3.03, 3.04, 3.05, 3.06, 3.07, and 3.08. Any other version of MacClade, such as 2.65, 2.87, 2.97, 2.99, 3 Beta 9.9, 4.0b1, and 4.0b22, were pre-release test versions that we showed at meetings or distributed to a few colleagues to test for us. (The easiest way to determine if a copy is a test version is to choose **About MacClade** under the  menu. If "Release Version" appears in the dialog box, or the version number is 1.0, 1.01, 2.1, 3.00-3.08, or 4.0, it is not a test version.) Test versions were distributed *for testing only*, under the condition that no results be published based on those versions without permission, as they may have bugs in many parts of the program, including the algorithms that give basic results such as tracing of character evolution and treelengths.

Unless we have given you express permission, DO NOT USE RESULTS FROM ANY TEST VERSION IN PUBLICATIONS. We expect that any analyses done with test versions will be redone with the officially released version 4 of MacClade.

Differences between MacClade 4 and 3

MacClade version 4 differs from version 3 in having a much more sophisticated data editor, a new system to manage characters, taxa, and trees (the list windows), and in various other features.

Data editing features have been improved in MacClade 4. Perhaps most significantly, there are many enhancements for dealing with molecular sequences. For instance, the cells of the matrix can be colored by state, amino acid translations can be shown alongside the nucleotide data, consensus sequences can be presented, and sequence alignment can be done both manually and automatically using the tools. The data editor now has a tool palette that includes tools that to some extent behave like those for a graphics program (e.g., fill data as if filling a color from a paint can). The data matrix can now be saved as an HTML file for presentation as a Web page. Printing of the matrix has been improved. NEXUS files can be better managed, as NEXUS blocks can be manually added and edited directly.

The tree window has a much enhanced Trace All Changes feature, which can color-code changes and temporarily highlight the reconstructed evolution of particular characters. Branch swapping is not only much faster, but includes the more powerful SPR algorithm (it is still no match for PAUP* in speed and effectiveness, but PAUP* cannot do searches using some of MacClade's assumptions, including the stratigraphic character type). MacClade has new features for interacting with PAUP*, in particular the automatic creation of PAUP* commands for such calculations as decay indices (Bremer support).

A broad change is the introduction of list windows, derived from MacClade 3's character status window. The list windows bring a unified interface to the management of items like characters, taxa, and trees. Along with the list windows, the menu structure of MacClade has been changed to emphasize these categories of items (characters, taxa, trees).

Differences in calculations or assumptions

Although some of MacClade's calculations of tree or character statistics have been reprogrammed since version 3 (for instance, to translate from assembly language to C), the algorithms used are the same in almost all cases and so should arrive at the same results as version 3. The exceptions are (1) the branch rearrangement algorithm used by the search tool, and (2) the algorithms for enumerating all most parsimonious reconstructions of character evolution (see "[Examination of all MPRs during MacClade's calculations](#)" on page 97). The latter algorithms are much more efficient than in version 3, and could give different results in rare circumstances, for MacClade 3's calculations could be in error when the number of MPRs is extremely high.

Future plans for MacClade

We plan to upgrade MacClade to maintain compatibility with future versions of the Mac OS. Although we have considered the possibility of adding substantial new features, such as likelihood calculations in the Tree Window, the additions are unlikely, as grafting major new features onto the existing MacClade source code would be difficult. Smaller changes are more likely. For example, we hope to increase the number of allowed characters, add DELTA format export, rebuild the pairwise alignment tool to use less memory, among other changes. We encourage users to send us suggestions for new features.

Although we plan to keep MacClade as an integrated and fairly compact program, we recognize that there are many analyses that could be performed that MacClade does not do. Thus, we are taking a dual strategy: maintaining MacClade with its strength in parsimony calculations and data editing, while at the same time developing a different system that emphasizes diversity of analyses instead of compactness. This new system is Mesquite, which is modular and designed to be extensible. As new modules are written by us or other programmers, they can be added to give new features. Mesquite is written in Java™ and will run on various operating systems. More information on Mesquite can be found at (<http://mesquite.bio-sci.arizona.edu/mesquite/mesquite.html>)

Registration

Make sure you register your copy of MacClade. The registration page is at

<http://www.sinauer.com/macclade/register.htm>

Electronic mail and World Wide Web addresses

For technical support (including bug reports) please use the electronic mail address clade@arizona.edu.

MacClade's World Wide Web site is at

<http://phylogeny.arizona.edu/macclade/macclade.html>

This site contains general information about MacClade, information about bugs, frequently asked questions and answers, and so on.

Reporting bugs

We have extensively checked and tested the main algorithms in MacClade in an attempt to ensure that no faulty results will be generated; in various parts of this manual we have described our checking procedures. We hope that no such bugs remain, but there will remain, we are sure, a number of bugs having to

do with screen display and occasional crashes. If you think you have found a bug in MacClade, please first check the list of known bugs at MacClade's World Wide Web site (at the URL listed in the preceding section). If you think your bug is a novel bug, or you are not sure, please report it to us.

To correct a bug, we need the following information:

1. A description of the problem. This includes a list of *all* the steps that led to the bug. If at all possible, try to repeat the problem yourself until you know the exact sequence of steps that reliably lead to the bug. Without knowing an exact sequence of events that reproduces the bug, there is a good chance we cannot find the bug and fix it.
2. We also may need an *exact* copy of the data files or tree files involved, as many bugs appear only with specific files. Therefore, a copy of the files should be saved. (If we do borrow your files, they will be kept strictly confidential, and will be discarded once the bug is fixed.)
3. Your name and address (e-mail if possible).
4. Exact version number of MacClade used (see **About MacClade** in the  menu).
5. Model of Macintosh.
6. Amount of memory (RAM) in your Macintosh.
7. Version number of the MacOS system (see **About Finder** or **About this Macintosh** or **About this Computer** in the  menu when in the Finder).
8. How much memory was allocated to MacClade (determined by selecting MacClade in the Finder and choosing **Get Info** and looking at the Memory panel).
9. Any other possibly relevant information about data file format (such as standard, extended standard, DNA, and so on), number of taxa and characters, type of printer, and so on.

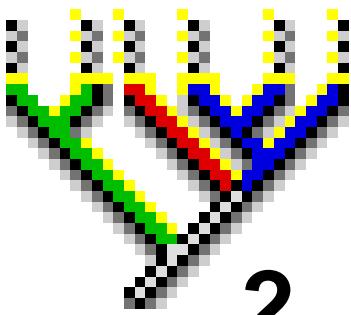
To report a bug, send the above information via electronic mail to clade@arizona.edu.

You may find that you are not sure whether you have discovered a bug or simply don't understand how MacClade is supposed to function. We would like to hear from you even if the problem is only that MacClade's intended behavior was not made clear to you. Before you contact us, however, please read the appropriate sections of this manual to see how MacClade was intended to behave.

Technical information and availability of source code

MacClade 4 was developed and compiled using Metrowerks CodeWarrior® Pro 4, based on the source code of MacClade 3 originally developed using Symantec's Think Pascal™. MacClade 4 is written primarily in Pascal, with some C code.

If you would like more technical information about MacClade, some details about the functioning of MacClade's algorithms are given in [Chapter 4](#) and the various other chapters concerning them. If you want further information on the algorithms or other technical details, please contact the authors. The source code for those parts of the program that deal directly with the calculations of results are available from us, so that they may be examined by the scientific community.



A TUTORIAL OVERVIEW OF MACCLADE

2

As an introduction to MacClade and some of its capabilities, we offer the following tutorial. It will guide you step-by-step through some simple data files. (We will assume basic familiarity with the operation of a Macintosh; if you need help with basic operations, see your Macintosh manual.) Those already familiar with MacClade may want to skip this chapter.

As you work through the tutorial, remember that the main windows in MacClade 4 are the data editor, the tree window, the chart window, and the list windows. The **data editor** is an editor in which the taxa are named, and their states in various characters are entered. In the **tree window** trees are built and modified, and their relationship to the character data explored. You can move between the data editor and tree window using items in the **Windows** menu. MacClade cannot show both windows at the same time; thus at any given time MacClade is either in data editing mode or tree analysis mode. When the tree window is on the screen, the **chart window** can be requested by choosing items from the **Chart** menu to show various summary statistics about the characters and their evolution. The list windows, including the **character list window**, the **taxon list window**, and the **tree list window**, allow you to manipulate and view information about characters, character sets, taxa, taxon sets, trees, and so on. For instance, the character list window lists the characters in the data file, and various assumptions about each character (its weight, assumptions about transformations from one state to another, and whether the character is currently included or excluded). This window not only lists these assumptions, but also allows you to change them. The character list window also lists statistics relating the characters to the current tree (e.g., their consistency indices) when the tree window is in use. By exploring these windows you will introduce yourself to many of MacClade's features.

Tree manipulation

Open the Examples folder, and within that the Tutorial folder. Double-click on the "Vertebrates" data file to start MacClade:

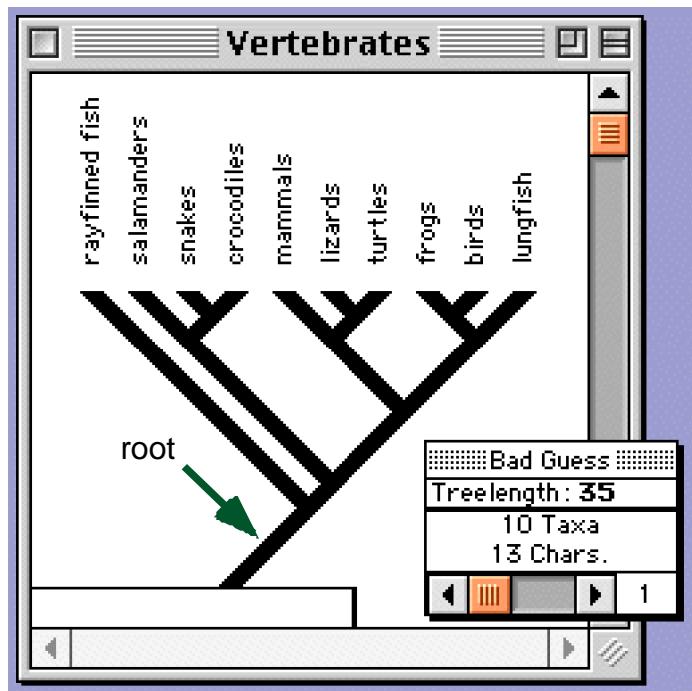


Vertebrates



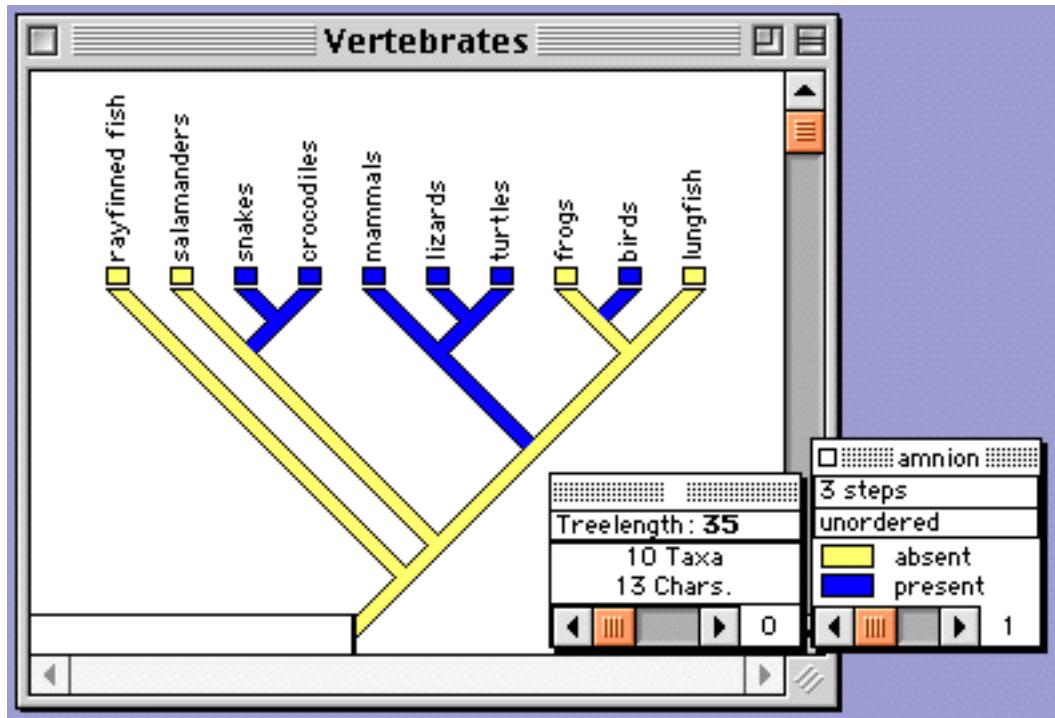
(In this document, when examples are being discussed, clicking on the file icon in the margin will also open a copy of the relevant data file; this will not work if you are viewing this file under Microsoft Windows [[page 111](#)], or if this file and the Examples folder are not in the same folder.)

The data file will take you straight to the tree window, and you will be presented with a phylogeny on the screen, as in the following figure. (If you are presented with a data matrix instead, the file has been altered — try selecting **Tree Window** from the **Windows** menu.)

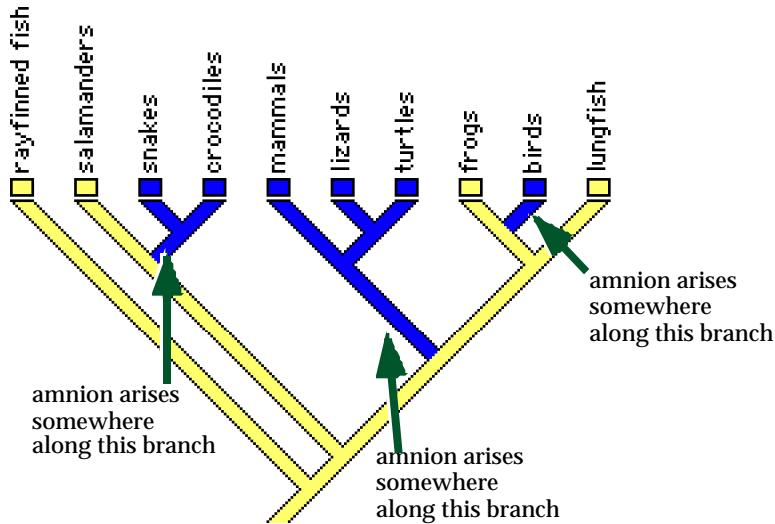


More accurately, what you see is one hypothesis about how the ancestral vertebrate lineage split and evolved, leading up to the major living vertebrate groups. As its name ("Bad Guess") implies, it is almost certainly wrong. The branch at the bottom of the tree, representing the ancestor of the taxa, is called the **root** of the tree.

To see how the traits of vertebrates have evolved according to this hypothesis, choose **Trace Character** from the **Trace** menu. You should now see the evolution of the amnion traced onto the screen.



Look at the legend in the lower right of the tree window. Note that the character amnion has two states: "absent" or "present". According to this phylogeny, the amnion (darkly shaded portions of the tree; blue if you have a color monitor) arose three times. This is also indicated in the legend at the far right, where it notes that there were "3 steps" of evolution in this character. These three origins are shown in the diagram below:

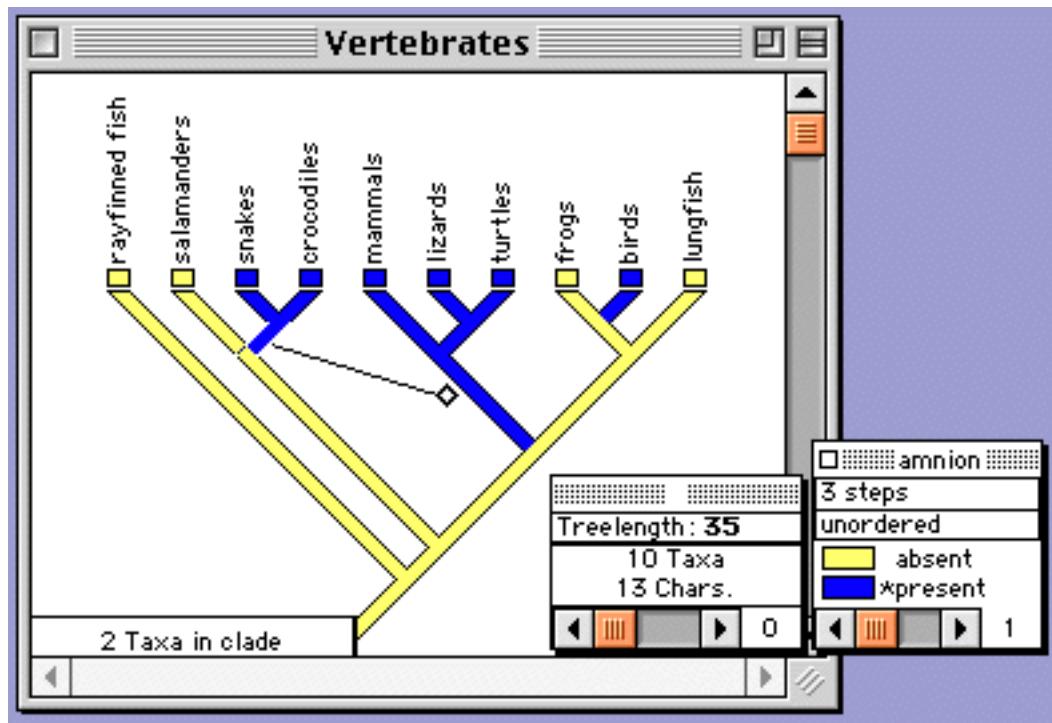


That the amnion arose three times independently (and thus that the amnions of snakes, mammals, and birds are **analogous**) seems unreasonable, as it is a complex explanation of the observations, and a simpler explanation is available. It seems more reasonable to presume that the amnion arose only once, in an ancestral tetrapod vertebrate, and that it was passed on to descendants of that ancestor (and thus that the amnions of snakes, mammals, and birds are **homologous**, having been derived from a common ancestor).

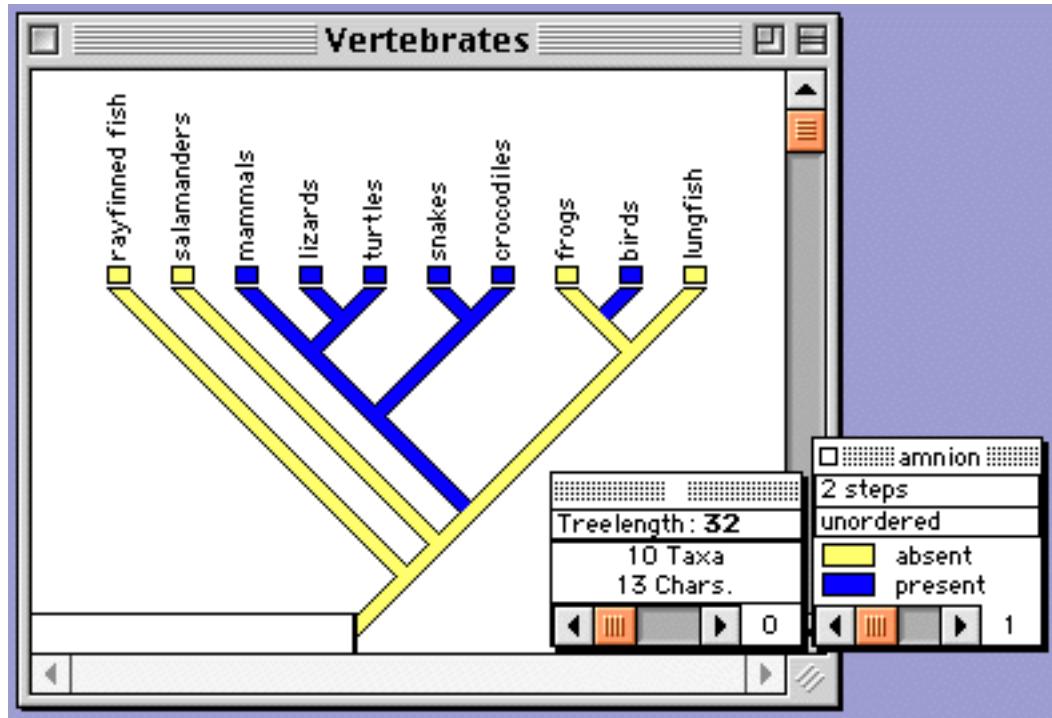
NOTE: Here MacClade shows the amnion arising at the start of a branch. MacClade does this for convenience only. The evolution of the trait might have happened anywhere along the history of the branch.

Now, let's alter the phylogenetic hypothesis slightly. Move the snakes + crocodiles over so that they now are next to the mammals + lizards + turtles. To do this, move the mouse's arrow pointer over to the branch

just below the snakes + crocodiles, click down on the mouse, and, while holding the mouse button down, drag the diamond over to the branch just below mammals + lizards + turtles:



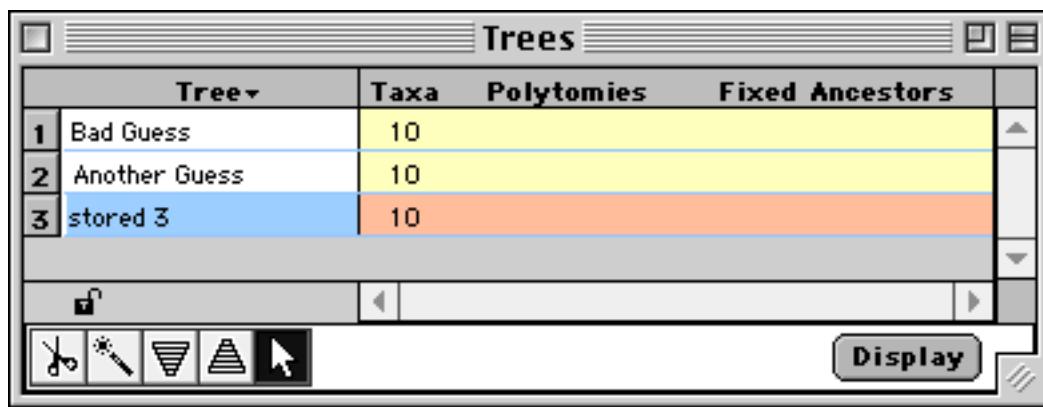
When the diamond is over the latter branch (a branch will change shades as the diamond passes over it), let go of the mouse button. The snakes and crocodiles should have moved over beside the mammals + lizards + turtles.



According to this new hypothesis, the amnion arose only twice; that's better, as it is a simpler explanation.

In the second box from the right is a line that reads "Treelength: 32". The **treelength** is a statistic that gives an indication as to how well the tree fits the data. The longer the treelength, the worse the fit; the shorter the treelength, the better the fit. The treelength is simply the sum, over all characters, of the number of evolutionary steps in each character on the tree. For the tree on the screen, the number of steps in the first character is 2; the number of steps in the second character is 3. (To scroll to see the second character, click once on the lower, right-hand arrow in the character legend at the far right.) In the third character, the number of steps is 5; and so on. The treelength is thus $2 + 3 + 5 + \dots = 32$. Shorter treelengths mean that, in general, the tree fits the characters well, that is, that the tree does not require one to presume multiple independent origins of traits.

Once you have found a tree you would like to save, store it in the file with the **Store Tree** command of the **Trees** menu. In the window that appears, the tree list window, you will see a list of stored trees:



The last item in the list is the tree you've just requested to store; it will be named "stored 3". The name is selected for you, and you can rename it by typing in a new name for it. You can reexamine the stored tree when you want by returning to this tree list window (available as the first item in the **Trees** menu), selecting the tree to display, and touching the Display button. (You may want to save the modified file, with its new stored tree, to disk so that you can see your tree the next time you open the "Vertebrates" file. You would do this using the **Save File** command of the **File** menu. However, if you do save the file to disk, it will no longer be in its original condition, making it more difficult for others to follow this tutorial.)

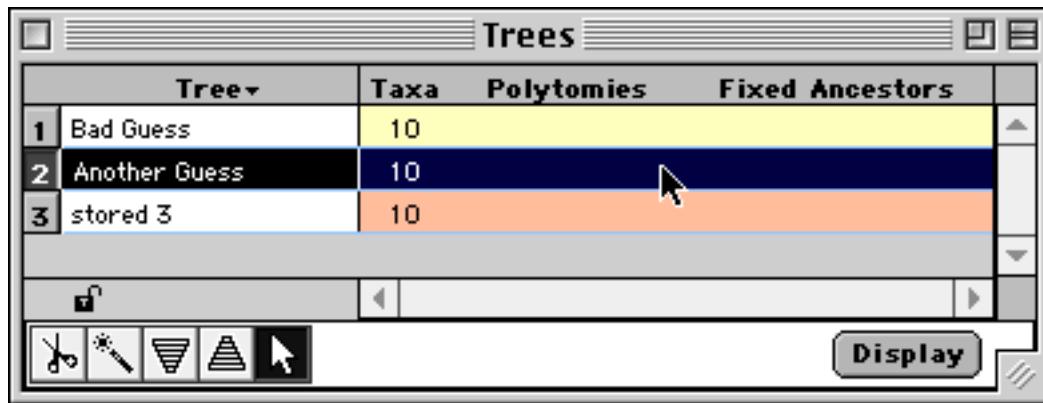
To see other options for tree manipulation, examine the tool palette (if the palette is not visible, select **Tool Palette** in the **Windows** menu). The small icons represent tools for your use. Choose the tool that looks like

this: , then click this tool on a branch in the tree. The tree will be rerooted where you touched it (unless you touched the tree at or near the root, in which case rerooting wouldn't make any difference).

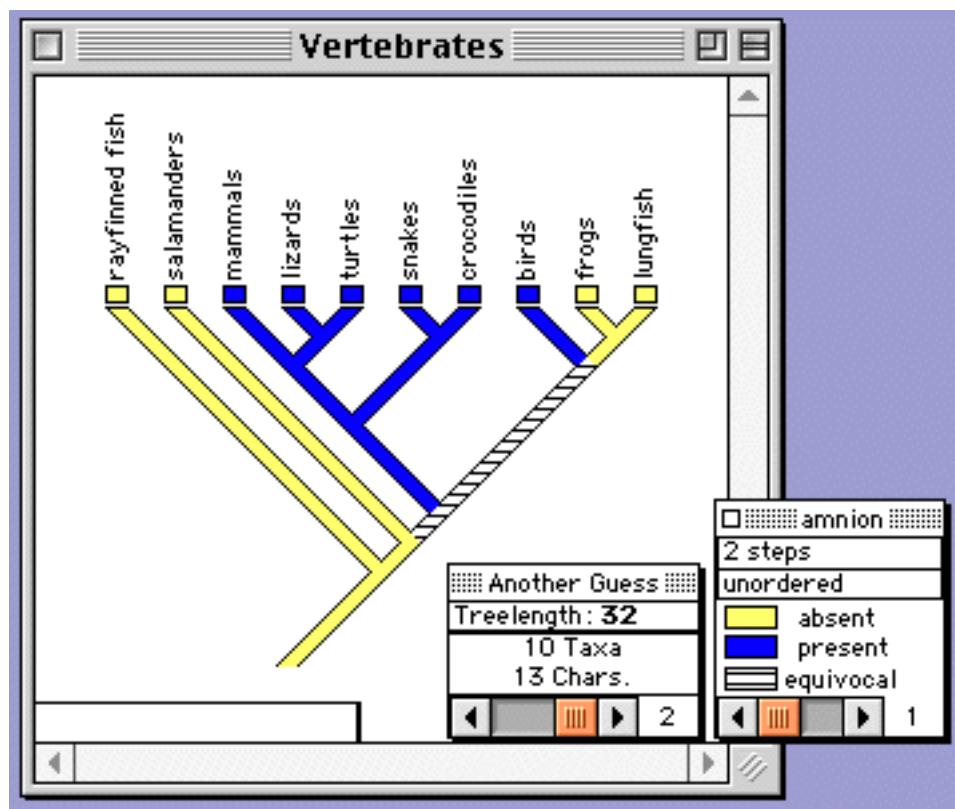
Try some other tree manipulation tools. will ask MacClade to try rearranging branches in the clade above the branch touched, in order to search for alternative trees that are more parsimonious according to

the data set. will destroy the branch touched and so yield a polytomy from a dichotomy. Play with the other tools to see what they do. MacClade lists the function of the tool in the small rectangle at the bottom of the palette. See [Chapter 17](#) for a more detailed description of these tools.

Once you have finished exploring the tree manipulation tools, open the tree list window (available by choosing **Tree List** in the **Trees** menu). Select the tree "Another Guess" by touching on the yellow part of its row, as shown below:



Then, press the Display button to have that tree displayed in the tree window. Trace character 1 onto the tree by scrolling to it using the arrows in the character legend. You should see the following:



Of course, if you were really serious about examining the evolution of this character, you would do it with a phylogenetic tree that was well supported by available evidence (clearly this one is not!). But this example is just for practice, so bear with us.

For the branches with horizontal striping (the "equivocal" pattern), the evidence is ambiguous as to how

the characteristics evolved. The ancestors along these branches might have had amnions, but they might not have; either possibility is equally parsimonious. When you click on a striped branch (do this now), MacClade indicates this uncertainty by placing a question mark in the legend box on the right beside each state that could be along that branch.

In this case, there are two possible patterns of character evolution; that is, two patterns that equally simply explain the distribution of the amnion in living vertebrates. They are:

1. The amnion arose independently in the mammals-to-crocodiles clade and the birds;
2. The amnion arose only once, but was later lost in frogs and lungfish.

To see these two possibilities, you can use one of the several tools MacClade provides for manipulating and examining the tracing. These are available in the tool palette. Select the paintbrush tool (). Click the paintbrush tool on the box in the legend to the left of "present", as shown in the following figure.



The paintbrush should turn colored (, indicating that it "contains" your selected state. (If you have a color monitor, and colors are used to indicate the states, then the paint brush will turn the color of the state; otherwise, a filled paintbrush will always be black, an empty one white.) Now click on one of the equivocal branches. In doing this, you are fixing the state of the branch; you are telling MacClade: "Show me the evolution of the amnion assuming that organisms along the branch I touched had an amnion". You should see the amnion arising once and being lost once. Now, undo what you just did by choosing **Unfix All** from the **Trace** menu. Next, click the paintbrush tool on the box in the legend to the left of "absent". Now, when you click the paintbrush on the same branch, you should see the amnion arising twice.

There are many other features to help you analyze character evolution, which we will introduce in other chapters (starting with [Chapter 18](#)).

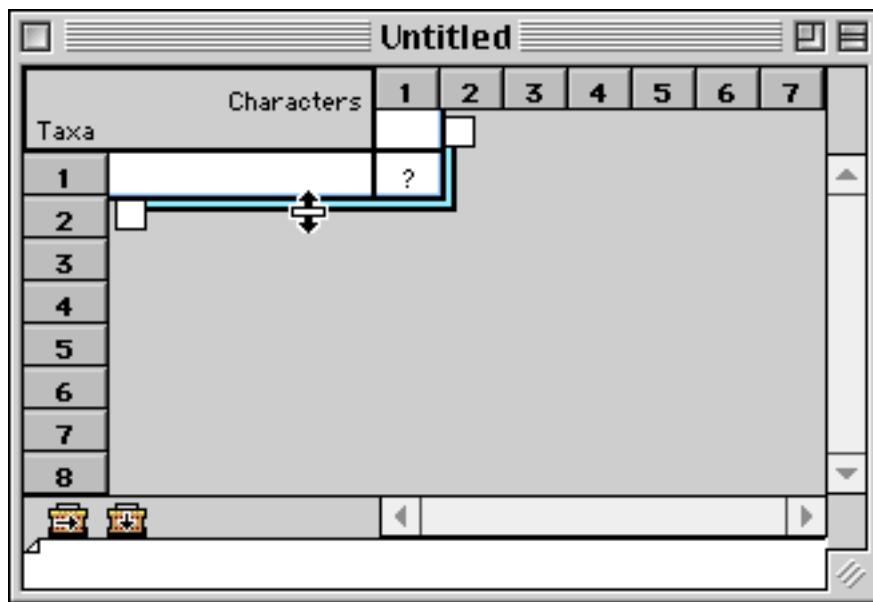
Entering data

Now let's examine the data that you have been analyzing. Go to the **Windows** menu and choose **Data Editor**. This will take you to a data editor that should look something like this:

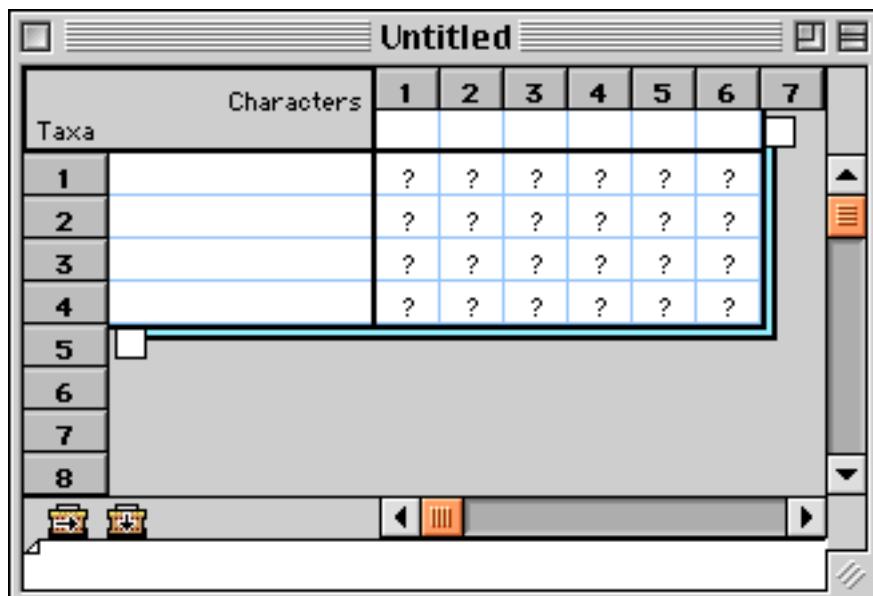
| Taxa | | Characters | 1 | 2 | 3 | 4 |
|------|----------------|------------|---------|-----------------|---------------|--------------|
| | | | amnion | appendages | body covering | thermoreq. |
| 1 | rayfinned fish | | absent | fins | derm.scales | poikilotherm |
| 2 | frogs | | absent | legs only | smooth | poikilotherm |
| 3 | turtles | | present | legs only | epid.scales | poikilotherm |
| 4 | lungfish | | absent | fins | derm.scales | poikilotherm |
| 5 | salamanders | | absent | legs only | smooth | poikilotherm |
| 6 | crocodiles | | present | legs only | epid.scales | poikilotherm |
| 7 | lizards | | present | legs only | epid.scales | poikilotherm |
| 8 | birds | | present | legs+wings | feathers | homeotherm |
| 9 | mammals | | present | s only/legs+wir | hair | homeotherm |
| 10 | snakes | | present | ? | epid.scales | poikilotherm |
| 11 | | | | | | |

On the left of each row are the names of the taxa. Along the top of each column are the names of the characters, and in the matrix itself are the names of the character states possessed by each of the taxa for each of the characters. If you want to change the data, click on the cell of the matrix you want to edit, and type in the new data. For instance, click on the cell in the first column (amnion) beside "salamanders" and type "present" if you want to claim that salamanders have an amnion.

Now let us make a new data file. In the **File** menu choose **Close File**. If MacClade asks you "Do you want to save changes before closing?", respond "No". MacClade will then present you with a list of files you might open. Instead of opening any of them, choose the New button to start a new data file. You will be presented with a blank data matrix with one taxon and one character:



The thick bluish border with a box at each end represents the edge of the matrix. To enlarge the matrix so as to add taxa, place the cursor over the bottom edge of this heavy border, as shown above (note the cursor), click and drag down. When you let go, MacClade will add more taxa. To make new characters, place the cursor over the right-hand heavy border, click and drag to the right. Add 5 characters, and 3 taxa. Your matrix should now look like this:



Now fill in taxon names in the wide column at the left-hand side. To enter the name of the first taxon, click on the cell beside the number 1, and type in the name:

| | | Characters | | | | | | |
|------|-------------|------------|---|---|---|---|---|---|
| | | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| Taxa | | | | | | | | |
| 1 | first taxon | ? | ? | ? | ? | ? | ? | |
| 2 | | ? | ? | ? | ? | ? | ? | |
| 3 | | ? | ? | ? | ? | ? | ? | |
| 4 | | ? | ? | ? | ? | ? | ? | |
| 5 | | | | | | | | |
| 6 | | | | | | | | |
| 7 | | | | | | | | |
| 8 | | | | | | | | |

Continue for the other taxon names. Now enter character state data in the body of the matrix. Enter character state data as 0s, 1s, 2s, and so on, by touching on a cell and typing the state number:

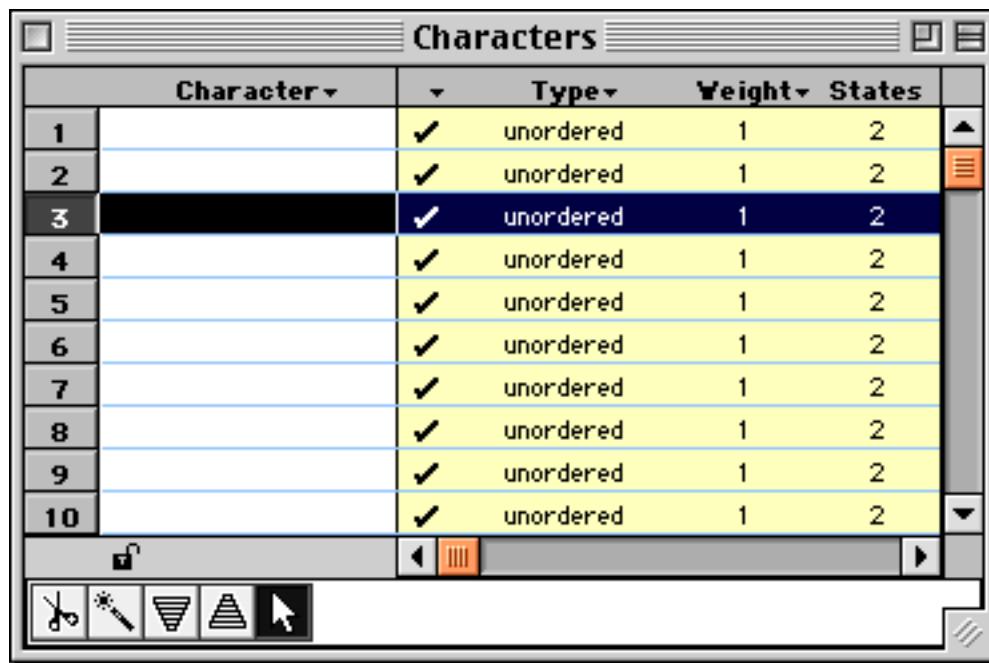
| | | Characters | | | | | | |
|------|-------------|------------|---|---|---|---|---|---|
| | | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| Taxa | | | | | | | | |
| 1 | first taxon | 0 | 1 | ? | ? | ? | ? | |
| 2 | another | 0 | 2 | ? | ? | ? | ? | |
| 3 | bird | 1 | ? | ? | ? | ? | ? | |
| 4 | mammal | 1 | ? | ? | ? | ? | ? | |
| 5 | | | | | | | | |
| 6 | | | | | | | | |
| 7 | | | | | | | | |
| 8 | | | | | | | | |

For now, we will name the states 0, 1, 2, and so on. If you want to use names like "absent", "present", "red", "blue", and so on in the matrix, refer to [Chapter 12](#) to learn how to prepare the data file for entering character state data by full names, as we did in the "Vertebrates" example, or for using other symbols such as ACGT. [Chapter 13](#) and [Chapter 14](#) explains many features of the editor not included in this Tutorial.

Specifying assumptions

Once you have typed some data into the new matrix, you can set the assumptions used for the characters. Choose **Character List** from the **Characters** menu. In this window are listed the characters and information

about them:



The screenshot shows the 'Characters' window in MacClade. The window title is 'Characters'. It contains a table with 10 rows, each representing a character. The columns are labeled 'Character', 'Type', 'Weight', and 'States'. The 'Type' column for all characters shows a checkmark next to 'unordered'. The 'States' column for all characters shows '1' and '2'. Row 3 is highlighted with a dark gray background, indicating it is selected. The bottom of the window has a toolbar with icons for various operations like cutting, pasting, and selecting.

| Character | Type | Weight | States |
|-----------|-----------|--------|--------|
| 1 | unchecked | 1 | 2 |
| 2 | unchecked | 1 | 2 |
| 3 | checked | 1 | 2 |
| 4 | unchecked | 1 | 2 |
| 5 | unchecked | 1 | 2 |
| 6 | unchecked | 1 | 2 |
| 7 | unchecked | 1 | 2 |
| 8 | unchecked | 1 | 2 |
| 9 | unchecked | 1 | 2 |
| 10 | unchecked | 1 | 2 |

To practice changing assumptions, click on the row for character 3 (click on either the row number at the far left, or in the yellow region of the row, *not* in the white region for naming characters), then go to the **Change Type** submenu in the **Characters** menu and select **Ordered**. This will change the type of character 3 from unordered to ordered. (The difference between an unordered and ordered character is described in [Chapter 4](#).) Now try excluding character 5 from subsequent analyses. To do this, select character 5 in this window. Then choose **Exclude** from the **Include-Exclude Characters** submenu in the **Characters** menu. The check mark will change to an x and the character type will be listed as "(excluded)". Character 5 is now excluded from any tree analysis.

Now try going to the tree window (using the **Tree Window** item in the **Windows** menu) and playing around with trees. (Don't forget to give names to all of the taxa first!)

Manipulating molecular sequence data

To see one of the tools provided by MacClade's data editor for manipulating molecular sequence data, open the file "DNA Data" in the Tutorial folder. This file contains data on a protein-coding gene in several



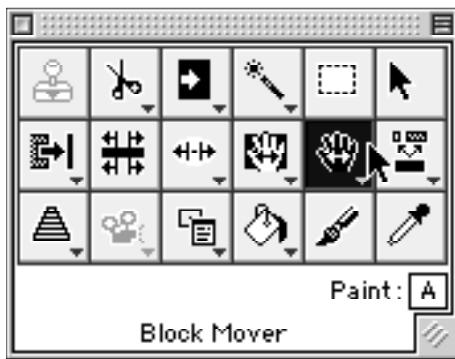
mammals, from Thomas et al. (1989). You will be presented with the data matrix, with the cells colored to make the nucleotides more obvious:

| | | Characters | 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 | |
|------|-------------|------------|---|---|---|---|---|---|---|---|---|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|
| Taxa | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | Thylacinus | - | - | - | C | T | T | T | G | G | A | T | C | C | T | T | A | C | T | A | G | G | A | A | T | C | T | G | C | | |
| 2 | Sarcophilus | - | - | - | C | T | T | C | G | G | G | T | C | T | T | T | A | T | T | A | T | A | G | G | A | A | T | A | T | G | C |
| 3 | Dasyurus | - | C | T | T | T | G | G | A | T | C | Y | C | T | A | T | T | A | G | G | A | G | T | A | T | G | C | C | T | | |
| 4 | Echymipera | - | - | T | T | T | T | G | G | G | C | T | C | A | T | T | A | C | T | A | G | G | A | A | T | C | T | G | C | | |
| 5 | Trichosurus | - | - | - | C | T | T | C | G | G | A | T | C | A | C | T | A | C | T | A | G | G | C | A | T | C | T | G | C | | |
| 6 | Phalanger | - | - | - | C | T | T | T | G | G | A | T | C | A | C | T | A | T | T | A | G | G | T | T | T | C | T | G | C | | |
| 7 | Philander | - | - | - | T | T | T | T | G | G | T | T | C | A | C | T | T | C | T | A | G | G | A | A | T | A | T | G | C | | |
| 8 | Bos | - | - | - | T | T | T | C | G | G | T | T | C | C | C | T | C | C | T | G | G | G | A | A | T | C | T | G | C | | |
| 9 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

A quick glance at these data suggest that the alignment of the sequences one to another is not ideal. This will be more obvious if you go to the **Display** menu and choose **Nucleotide with AA Colors** from the **Data Matrix Styles** submenu. This will ask MacClade to color the cells not by the nucleotide state, but by the amino acid into which each nucleotide triplet would be translated, given the specified reading frame:

| | | Characters | 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 | |
|------|-------------|------------|---|---|---|---|---|---|---|---|---|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|
| Taxa | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | Thylacinus | - | - | - | C | T | T | T | G | G | A | T | C | C | T | T | A | C | T | A | G | G | A | A | T | C | T | G | C | | |
| 2 | Sarcophilus | - | - | - | C | T | T | C | G | G | G | T | C | T | T | T | A | T | T | A | T | A | G | G | A | A | T | A | T | G | C |
| 3 | Dasyurus | - | C | T | T | T | G | G | A | T | C | Y | C | T | A | T | T | A | G | G | A | G | T | A | T | G | C | C | T | | |
| 4 | Echymipera | - | - | T | T | T | T | G | G | G | C | T | C | A | T | T | A | C | T | A | G | G | A | A | T | C | T | G | C | | |
| 5 | Trichosurus | - | - | - | C | T | T | C | G | G | A | T | C | A | C | T | A | C | T | A | G | G | C | A | T | C | T | G | C | | |
| 6 | Phalanger | - | - | - | C | T | T | T | G | G | A | T | C | A | C | T | A | T | T | A | G | G | T | T | T | C | T | G | C | | |
| 7 | Philander | - | - | - | T | T | T | T | G | G | T | T | C | A | C | T | T | C | T | A | G | G | A | A | T | A | T | G | C | | |
| 8 | Bos | - | - | - | T | T | T | C | G | G | T | T | C | C | C | T | C | C | T | G | G | G | A | A | T | C | T | G | C | | |
| 9 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

To realign the Dasyurus sequence, choose the Block Mover tool () from the tool palette:



Grab the Dasyurus sequence and move it slightly to the right, until the amino acids align more closely with those of Sarcophilus:



| | | | | | |
|----------|-------------|-----|----------------------------|----------------------|----------------|
| 2 | Sarcophilus | --- | CTTCGGGTCT | TTTATTAGGAATATG | GCCTAATTATTCAA |
| 3 | Dasyurus | --- | CTTGGAATCYCTATT | AAGTATGCCTAATTATTCAA | |
| 4 | Echymipera | -- | TTTGGCTCATTA | GAATCTGCT | TCATCCAAA |
| 5 | Trichosurus | --- | CTTCGGATCACTACTAGGCATCTGCT | TTAACTATAACAA | |

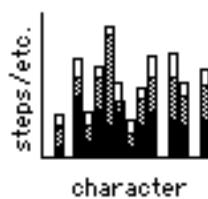
Although you could do the same to align Echymipera, instead let's use MacClade's pairwise alignment tool (). Choose it from the palette, touch on the name Echymipera, and continue to hold the mouse button down as you drag the Echymipera sequence onto the Dasyurus sequence. When you let the mouse button go while over the Dasyurus sequence, the sequence dragged (Echymipera) will be aligned to the sequence onto which it was dropped (Dasyurus).

Charting results

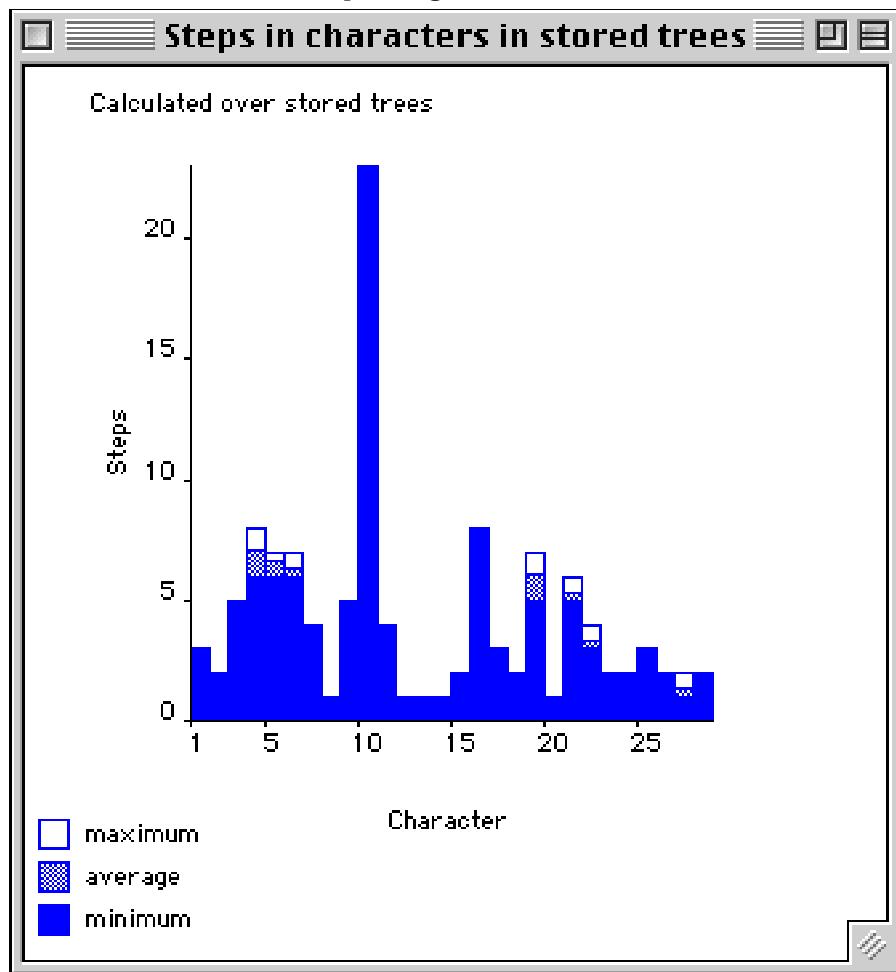
We can now briefly explore some of MacClade's charting facilities.



Close the current file, and open the file "Amblygnathus" in the Tutorial folder. This file contains morphological data on a group of ground beetles (Ball and Maddison, 1987). Go to the tree window, and choose **Character Steps/etc.** from the **Chart** menu. In the dialog box that appears, there are six large icons; touch on the lower-middle icon:



When you press Chart, MacClade will present you with a chart summarizing the number of steps in each character over the 21 most-parsimonious trees that are stored in the data file. The 28 characters will appear along the horizontal axis, the number of steps along the vertical axis:

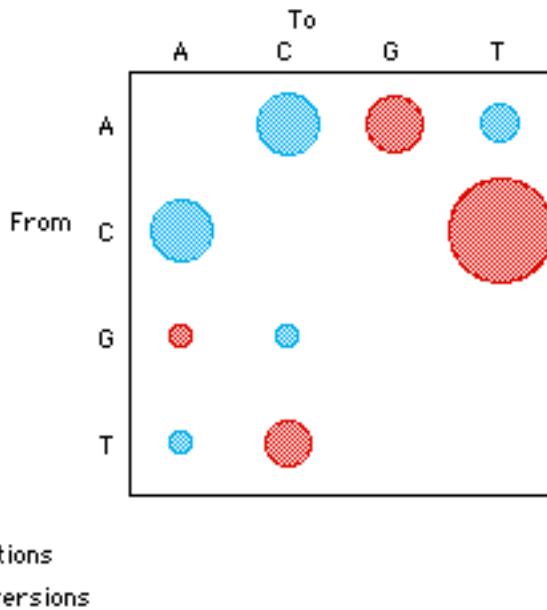


In the chart that appears, most of the bars are solidly filled (with black or a color, depending on your monitor), indicating that those characters have the same number of steps on all 21 trees examined. For seven of the characters (4, 5, 6, 19, 21, 22, and 27), however, part of each bar is solid, part is grayed, and the top is white. The number of steps in these characters varies over the 21 trees. The solid part of the bar indicates the minimal number of steps across trees, the gray part the average number, and the white part the maximum number. For example, the chart indicates that character 4 has between 6–8 steps, with an average number of 7 steps.



To explore another sort of chart, close this file, and open the file "Marsupial Wolf". This file contains DNA sequence data for the cytochrome b gene for various marsupials (Thomas et al., 1989). In the tree window, choose **State Changes & Stasis** from the **Chart** menu. In the dialog box that appears, press Chart. This asks MacClade to make a chart of the relative frequencies of each kind of nucleotide change for the DNA sequence data over the tree on the screen, as reconstructed by parsimony. The chart presented should look something like the one shown below. Note the relative abundance of C to T changes.

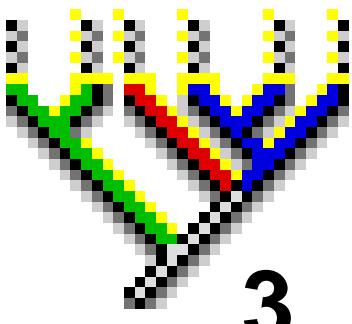
Frequency of unambiguous changes between states in tree Fig. 2 tree



Concluding remarks

You have now seen some of the major parts of MacClade. Other features that you may quickly need include printing ([Chapter 25](#)), and the ability to custom-make your own assumptions about how character states evolve one to another ([Chapter 15](#)). Examples of some features useful for molecular data are presented in [Chapter 16](#).

NOTE: As you explore MacClade, be aware of the power of the *Option* key. When you hold down the *Option* key, menus and tools may change to give you access to features not otherwise available. These features are described in appropriate places throughout this manual.



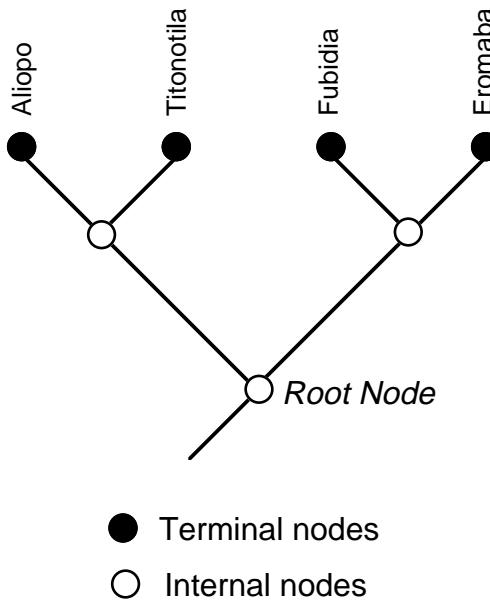
OVERVIEW OF PHYLOGENETIC INFERENCE

3

MacClade's focus is inferring evolutionary history and understanding its implications. In this chapter we provide an introduction to relevant terms and a theoretical background.

Trees and their terms

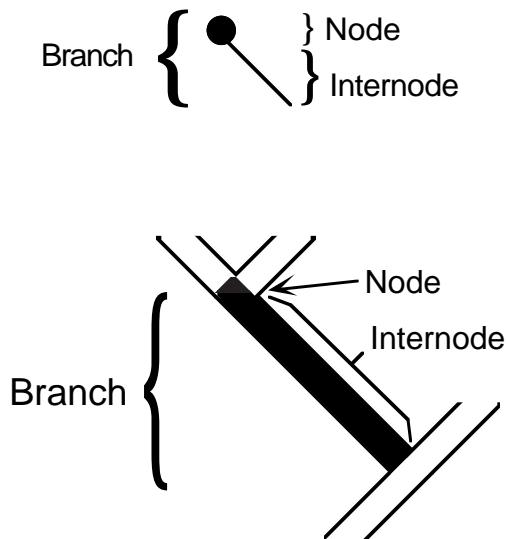
A diagram depicting a phylogenetic tree has a simple form of branching lineages. The point at which a lineage branches or ends is referred to as a **node** (also sometimes called a vertex). The nodes at the tips of branches have no descendants—these are the **terminal nodes**. They correspond to entities (like species) treated as indivisible for the purpose of the analysis.



Traditionally, trees have represented the relationships among species, with the terminal nodes representing observed species or clades of species. The precise interpretation of such nodes would therefore depend on the species concept used. Increasingly, however, trees are being used in studies of gene descent, with the terminal nodes representing individual gene copies. Nodes not at the tips of the branches are **internal nodes**; they generally refer to unobserved (hypothetical) lineages, unless a claim is being made that an observed taxon is an actual ancestor. Between the nodes are the **internodes** or edges. A tree is therefore composed of nodes and internodes. A tree is rooted if it has a node that is designated as the **root**. The root represents the ancestor of the tree. With the root so designated, the tree has polarity, and one can speak of nodes being **ancestors** or **descendants** of one another. In this manual we will sometimes refer to nodes above or below other nodes, meaning nodes descended or ancestral, respectively, following MacClade's usual tree orientation with time proceeding from bottom to top. In MacClade, the term "root" is used specifically for the internal node that is the most recent common ancestor of all other nodes in the tree, as shown above. As shown in the diagram above, MacClade draws an internode dangling below the root even though there is no node at its bottom end. This dangling internode is drawn to yield a larger object for ease of manipulating the root node and also to suggest the tree's connection with the rest of the

tree of life.

Nodes always have only one immediate ancestor unless there is reticulation or hybridization. (Reticulate trees cannot be directly represented in MacClade.) In a fully **dichotomous** tree each node has exactly two descendants. A **polytomous** node has more than two descendent nodes. There are at least two interpretations of a polytomous node (W. Maddison, 1989; see [Chapter 17](#)), one in which it represents uncertainty in relationships and another in which it represents multiple simultaneous branching.



In MacClade the term **branch** is used to refer to a node plus the internode immediately below it. In MacClade's graphical style this would be drawn as shown in the lower figure.

Because it is difficult to display graphically all of the required information about a node in the small region of the node itself, MacClade sometimes uses the whole branch to display information that actually refers to the node at the top of the branch. For instance, in tracing the evolution of a character, MacClade shades the whole branch to indicate what character states are reconstructed at the node at the top of the branch (see the section "[Interpreting the character tracing](#)" on page 345). The user must be aware of such distinctions: although MacClade treats the branch as a single graphical unit, the information displayed on the branch sometimes refers to the node at the top of the branch (e.g., Trace Character's indication of states on a node) and sometimes refers to the internode of the branch (e.g., Trace All Changes' indication of changes that occur along the internode).

A **monophyletic group**, called a **clade**, consists of an ancestor and all of its descendants. Because our collections from tropical forests are still incomplete and time machines are not yet available, we can never have in front of us *all* of the descendants of a common ancestor. Thus when a claim is made that a recognized group, for example "birds", is monophyletic, it means that there exists a clade of which the known birds represent the only and all of the sampled members. Usually, it is convenient to use a qualified concept of monophyly that restricts itself to the sample of organisms under consideration (e.g., the set of taxa in your data file): a group of these organisms is monophyletic if it has a single most recent common ancestor that is not also an ancestor of organisms in the sample that are not included in the group. This definition is more useful than the one that began this paragraph not only because it is applicable to samples of organisms. It also avoids complications within species by disallowing the designation "monophyletic" for groups whose most recent common ancestors are not unique to them, as can happen in phylogenies with reticulations (for instance within species; de Queiroz and Donoghue, 1990). In terms of nodes on a tree diagram, a **clade** includes a node and all nodes descendent from it.

The **ingroup** is a set of taxa, often assumed monophyletic, designated as being the focus of interest, as

compared to the **outgroups**, which are brought into the analysis to provide a broader phylogenetic context to aid in determining the root of the ingroup or ancestral states (Farris, 1972, 1982; Watrous and Wheeler, 1981; Maddison et al., 1984). In MacClade no distinction is made between ingroup and outgroups; it is up to you to formulate and maintain your own distinctions as you work with the program. A **taxon** is one or a group of species (or genes, or other entities), and might be either a clade in your tree, or a taxonomic unit placed in terminal position on the tree (a **terminal taxon**, or "Operational Taxonomic Unit"). As should be clear from context, sometimes we will use the term "taxon" to refer specifically to a terminal taxon.

Not all authors would agree precisely with our definitions of terms concerning trees, for the tree diagrams used throughout phylogenetic biology have had different meanings to different authors (for discussion see Nelson, 1974; Platnick, 1977; Hull, 1979; Eldredge and Cracraft, 1980; Nelson and Platnick, 1981; Wiley, 1981; W. Maddison, 1997). A branching diagram showing relationships among species in the form of a tree might be interpreted to represent:

1. *The hypothesized phylogenetic history, with lines representing lineages descending through time and branch points representing speciation events.* If the diagram places two observed species on separate terminal branches next to one another, this means that they share a common ancestor unique to them, and neither is the ancestor of the other. If one had been an ancestor, it would have been placed directly at an internal node of the tree diagram. Thus, the diagram depicts both recency of common ancestry and whether or not observed taxa are ancestors. Because this interpretation makes reference to "lineages" and "species", different species concepts would yield different variants on this interpretation (Mayr, 1963; de Queiroz and Donoghue, 1988; Nixon and Wheeler, 1990; Baum and Shaw, 1995; W. Maddison, 1997).
2. *The relative recency of common ancestry of the observed taxa.* This interpretation differs from interpretation 1 in that the tree diagram is silent on the issue as to whether any of the observed taxa might be ancestors of other observed taxa. By placing several observed taxa in a clade, the diagram claims only that these taxa share a common ancestor not shared by other taxa outside the clade. Even though each observed taxon is placed in terminal position on the diagram, separated from the internal nodes by an internode, the internode leading up to the terminal node might or might not have existed in nature — no claim is made either way. That is, the diagram allows the possibility that the observed taxon might actually be the ancestral lineage represented by the internal node to which it is connected.
3. *The hierarchical distribution of shared, derived, homologous character states (synapomorphies).* A "clade" in the diagram represents not necessarily a group of taxa sharing uniquely a common ancestor, but rather a group of taxa united by concordant synapomorphies; the internode below the clade represents the evolutionary derivation of these synapomorphies. Although this interpretation does not speak of lineages and speciation, it is evolutionary insofar as it summarizes evolutionary changes.
4. *The hierarchical distribution of shared characteristics.* Various characteristics are more or less general in their distribution and it is presumed that their generalities form a nested hierarchy. This hierarchy might arise by branching evolution, or by the laws of ontogeny, or by some other cause. Placing taxa in the same "clade" does not necessarily mean that they share an evolutionary innovation, but rather that they share a similarity. This interpretation makes no claims about evolution.

How are MacClade's tree diagrams to be interpreted? Although you may be able to use MacClade regardless of which of the four interpretations you prefer, our perspective (and MacClade's) is very much one of lineages, ancestors, and branching. As such, the appropriate interpretations are 1 or 2. Interpretations 3 and 4 are less common in recent literature than they were in the past.

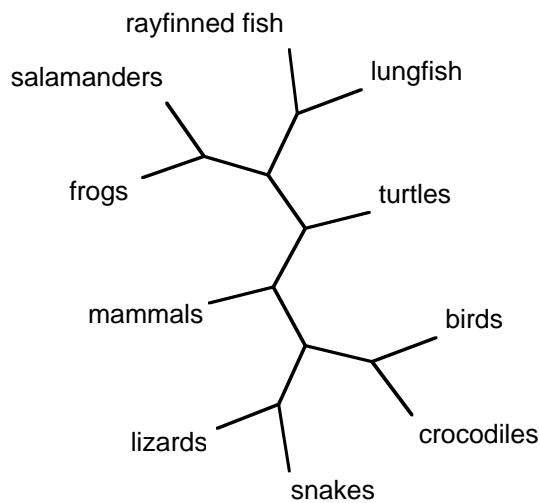
The distinction between interpretations 1 (recency of common ancestry and ancestral status) and 2 (recency of common ancestry only) has been considered an important distinction (Nelson, 1974; Platnick, 1977), because the difficulty of finding evidence that a species is ancestral leads some authors to prefer

remaining silent on the ancestral status of species. However important this distinction may be, we do not view MacClade as dealing solely with one or the other interpretation. Indeed, in the context of MacClade it is difficult to maintain a crisp distinction. First of all, a diagram or hypothesis need not be completely specific nor completely silent on the issue of ancestry. Perhaps you know of derived characteristics for some species, and can therefore make strong claims that they are not ancestral, whereas for other species you have no evidence about their status as ancestors. A tree diagram might be annotated to indicate which species are claimed to be terminal, and which species might be either ancestral or terminal. Second, as soon as you indicate that an observed taxon is in ancestral position (as can be done in MacClade), the tree at least partially commits itself to claims about ancestral status. Such a claim might be supported by stratigraphic parsimony (Fisher, 1991, 1992). Third, and perhaps most common and most subtle, as soon as MacClade reconstructs a change on a terminal branch and you accept this reconstruction as a working hypothesis, then the tree can no longer be considered silent on the issue of ancestry. The change on the terminal branch implies that the terminal branch exists, that is, that the observed taxon is in fact terminal and not ancestral, and therefore it would be inconsistent both to accept the hypothesis of change and simultaneously to allow for the possibility that the taxon was ancestral. Fourth, various of MacClade's features, such as the concentrated-changes character-correlation test, and the random generation of characters on the tree, assume that changes can occur on terminal branches. If you use these features, you are assuming that the tree is fully specific about ancestral status, that is, that *all* observed taxa that are not explicitly placed as ancestors are actually terminal.

Various names have been given to these various interpretations of tree diagrams (Platnick, 1977; Eldredge and Cracraft, 1980; Wiley, 1981; Hendy and Penny, 1984), but because their usage varies we will tend to avoid making fine distinctions. The term "phylogenetic tree" has been used for interpretation 1, the fully specific hypothesis of branching of ancestors and descendants, but sometimes the term is used to refer to the history itself (not just our hypothesis of it). "Cladogram" has been used variously for interpretations 1, 2, 3, and 4, whereas "synapomorphy diagram" has been used for 3 and 4. To avoid misleading readers accustomed to a particular interpretation of these words, especially "cladogram", and for the sake of compactness, we will most commonly use the word "tree" when we refer to the structures displayed by MacClade. Occasionally we will use the term "phylogenetic trees" to refer to our hypothesis about phylogenetic history or even to refer to the history itself. We apologize to the reader for any confusion this may cause. It should be clear from the context whether we are referring to the history, the hypothesis, or either. As discussed, a "phylogenetic tree", as we use the term, may have some elements of both interpretations 1 and 2.

In some circumstances there will not be enough information provided to discover the node of a tree on which the root resides; that is, the location of the ancestor from which the group diversified will be unknown. One will then speak of an **unrooted tree**, and in representing the relationships, a diagram that does not specify the root may be shown. For example, one of the possible unrooted trees for the "Verte-

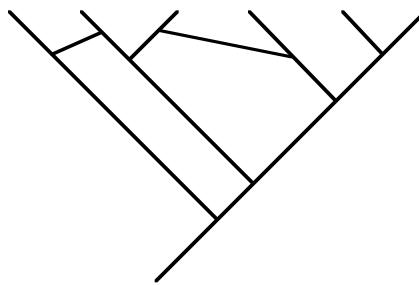
brates" example file is shown below.



An unrooted tree is no more than a type of consensus tree, a summary diagram of the many rooted trees that could be formed from it by specifying the location of the ancestor. The unrooted Vertebrate tree shown above is a summary diagram of 17 rooted trees that could be created by rooting the tree on each of the 17 branches.

To root a tree is to specify on it the polarity of time from past to present. Evolution, following time as it does, always produces rooted trees. It is our ignorance that produces unrooted trees. Although MacClade draws trees as if they are rooted, you may still interpret the trees as unrooted, with the root chosen merely as a graphical convenience.

Phylogenetic history can not always be represented as a single tree. When species hybridize, different portions of the genome will trace their ancestry through different parental species. Such a history can be summarized by a **reticulate phylogeny**, a simple example of which is shown below.



Reticulate phylogenies are necessary to describe the history of prokaryotes at least, because of the gene transfers that have occurred between different lineages (Doolittle, 1999). However, MacClade is not capable of directly considering reticulate phylogenies.

Reticulation is not the only process that makes it difficult to represent phylogenetic history as a single tree. Within sexually reproducing species, recombination causes different portions of the genome to have slightly different histories within populations, which can lead eventually to a diversity of gene histories which cannot all be summarized conveniently by a single species tree diagram. This problem, called incomplete lineage sorting or deep coalescence (Avise et al., 1983; Doyle, 1992; W. Maddison, 1997), may provoke further changes in the ways we reconstruct and interpret phylogenetic trees (W. Maddison, 1997).

Characters and their terms

A **character** can be viewed as a set of alternative conditions, called **character states**, that are considered able to evolve one to another. Thus the character "eye color" might consist of the states "eyes blue" and "eyes red", and it is assumed that red eyes can evolve into blue eyes and vice versa. A goal of phylogenetic analysis might be to discover how these different conditions might have evolved into one another — which is the ancestral condition, whether and where convergence has occurred, and so on. The decision as to what states can be considered part of the same character (i.e., what conditions represent transformations of one another; what conditions are homologous) may be made prior to a phylogenetic analysis, or it may be made simultaneously (Sankoff and Rousseau, 1975; Patterson, 1982). Simultaneous decisions about phylogeny and which features can be considered states of the same character are generally not made in computer phylogenetic analysis, as available programs usually require specification of states and characters beforehand. One exception is simultaneous sequence alignment and phylogeny reconstruction with molecular data (Sankoff and Rousseau, 1975; Hein, 1989).

Within the context of a particular group of organisms, an **apomorphy** is a character state derived within the group, whereas a **plesiomorphy** is the ancestral character state (the state at the most recent common ancestor of the group). A derived character state shared by members of a group is a **synapomorphy** of the members of the group, and an **autapomorphy** of the group. When a character state evolves more than once in different branches of the tree, there is **homoplasy**. That is, homoplasy is similarity that cannot be directly attributed to common ancestry. If the character has a state, then it changes, then reverses to its original state, the process is called **reversal**. If the state evolves separately on different branches of the phylogenetic tree (branches that are not in ancestor-descendant relationship), the process is called **convergence** or **parallelism**. The concept of homoplasy can be made slightly broader by saying that whenever a character evolved with more changes or more costly changes (in the sense of "cost" used in a parsimony analysis) than the minimum conceivable, then homoplasy is implied.

Coding character states

Before a collection of data, perhaps consisting of illustrations of mouth parts, leaf shapes, or molecular sequencing gels, can be used in computer-based analysis, the data need to be **coded** in a form acceptable to the computer program to be used. Coding the various forms of a character as states is a crucial but often under-appreciated step of phylogenetic analysis (Meacham, 1984; Pogue and Mickevich, 1990; Stevens, 1991). Although the coding of molecular sequence data in general is obvious ("A", "C", "G", and "T" for DNA data, for example), the appropriate coding for other characters may require difficult decisions. A character may naturally be **discrete-valued** ("red", "blue", "green"; "absent", "present"), or it may have gradations of value and be **continuous-valued** (0.197, 0.231, 0.224, 0.258). Even if it is a continuous character, it may be coded for convenience as if it were discrete, with certain ranges of values grouped together as one state (e.g., Archie, 1985; Stevens, 1991). Even when a character seems discrete, it may have states whose separation is not as clear as one would like. Decisions made in coding the character will have an effect on all later stages of phylogenetic analysis.

A clear distinction should be made between how states should be coded in theory, and what codings can be accepted by the available computer programs. For instance, before computer programs could accept multistate characters, characters were decomposed by additive binary recoding (Sokal and Sneath, 1963), often with unexpected consequences such as implicit ordering of the states. Before computer programs could accept polymorphic taxa, these may have been coded as having missing data (Nixon and Davis, 1991). Attempts to use certain assumptions about character evolution often required special coding methods when those assumptions were not directly available in the computer programs. For instance, additive recoding (e.g., O'Grady and Deets, 1987) is needed when programs can not directly handle character state trees. The "X-coding" of Doyle and Donoghue (1986) attempted to capture special assumptions about transformability between states, but now the assumptions can be approximated by step matrices (see below). Even now there are some situations that available computer programs do not directly handle, and recoding remains an important option. (Note for instance the discussion about ancestral polymorphisms

below and in [Chapter 4](#).) Please take the time to learn the full capabilities of MacClade before recoding your data: it can handle missing data, polymorphic or uncertain states, multistate characters with various assumptions about transformations, characters with as many as 26 discrete states, and, in a limited way, continuous-valued characters.

Polymorphisms and missing data

A given terminal taxon may be **monomorphic**, having a single state ("A"), or **polymorphic**, having multiple states ("A and C"), or of **uncertain** state ("A or C"), or its condition may be completely unknown (**missing data**). In MacClade, each of these codings is allowed, but it is important to consider certain dangers of coding polymorphisms and missing data.

Before you fill your data matrix with codings of terminal taxa having multiple states (polymorphisms), you should think carefully about the cautions given by Nixon and Davis (1991). Although they phrase their objection to polymorphisms in terms of missing data codings, their main point holds whether the polymorphism is coded as missing data or directly as polymorphism as can be done in MacClade. The potential difficulty arises when a terminal taxon is polymorphic in more than one character. If the terminal taxon were decomposed into monomorphic component parts, it might be seen that the state distributions of the two characters interact, such that some combinations of states are not parsimonious combinations for the most recent common ancestor of the terminal taxon under any potential arrangement of the component parts. Leaving the taxon intact but polymorphic hides the problems and allows the algorithms to use these combinations despite the fact that they are not parsimonious. This problem is another incarnation of the problem discussed by Maddison et al. (1984:96) concerning unresolved outgroups and by W. Maddison (1989) concerning the length of trees with soft polytomies. In every case, an unresolved region of the phylogeny is analyzed without adequately considering interactions between characters.

With terminal taxa the question is, Should one always avoid polymorphic codings and either infer an ancestral state for the taxon or decompose it and use the monomorphic component parts? Using an inferred ancestral state would avoid the problem, could the inference be well defended. Decomposing the taxon could also be done, but its success depends on how well the component parts are known. For instance, perhaps your knowledge that a genus is polymorphic comes from studies that were able to identify specimens only to genus. Decomposition could not then be done effectively. Success of decomposition depends also on whether you are prepared to take on the task of analyzing the components' phylogeny at the same time as the rest of your taxa. (What if the polymorphic terminal taxon was "Insecta"?) In some circumstances, you must simply admit that you have incomplete information and leave the taxon coded as polymorphic. After all, assuming one lacks precise information about the monomorphic components, there is a most parsimonious interpretation of evolution within polymorphic terminal taxa, and this is what MacClade allows. One can provisionally be satisfied with the parsimonious interpretation. Finally, decomposition may be inappropriate if reticulate evolution dominates within the taxon (for instance, with a species) and it is therefore not decomposable into monophyletic groups. Nonetheless, the points raised by Nixon and Davis (1991) do give one good reason to try to avoid polymorphic codings if possible.

A second problem with coding a character as polymorphic in a taxon has to do with limitations on the algorithms for reconstructing ancestral states. These algorithms assess alternative assignments of ancestral state to internal nodes. In most existing algorithms, including those in MacClade, the alternative assignments are mutually exclusive; that is, the internal node might have either state "yellow" or state "red", but not both at the same time. This prohibits ancestors from being polymorphic.

As described in [Chapter 4](#), this restriction can be avoided if the character is coded so that its states refer to species and not to individuals (see Mickevich and Mitter, 1981; Farris, 1983:27). "Eye color yellow" is a character state of a salamander, not of a population or species of salamanders; "All individuals with yellow eyes" could be the character of the species. The character referring to species states would have three states, all yellow, all red, and polymorphic. By using characters coded at the species level, the algorithms can allow ancestors to be polymorphic.

Is MacClade's ability to code a character as polymorphic ("0&1") in a taxon in the data matrix therefore not useful? We have just discussed how a polymorphic coding might be better decomposed into components, or that a polymorphic coding should be changed to a single-state coding reflecting the species-level condition of being polymorphic. However, there are reasons not to decompose or recode polymorphisms. If the terminal taxon is a higher taxon and the "polymorphism" amounts to uncertainty about the terminal taxon's ancestral state, then decomposition may be impractical and recoding inappropriate. If the terminal taxon is a species, then recoding may be impractical (too many states, step-matrix calculations too slow) and decomposition inappropriate. We will therefore often be forced to code the characters in a less-than-ideal way. At the very least, coding polymorphic taxa as polymorphic ("0&1") is better than coding them as having missing data. Finally, it may be best to save a copy of the data matrix with polymorphisms listed by organism states so that the data file might be better used when better algorithms come out in the future that do allow ancestral polymorphisms.

Coding a taxon as having missing data in a character should also be done with caution. The character state possessed by a taxon may be unavailable either because the state is not known, or because the character is simply inapplicable to the taxon (e.g., feather type in a crocodile; Platnick et al., 1991; Nixon and Davis, 1991). The missing data coding in MacClade is intended for the former case, when the state is simply unknown. Coding an organism as having missing data because the character does not apply to the organism can lead to implications of impossible ancestral states (Platnick et al., 1991; W. Maddison, 1993). This can lead to an unjustified choice of trees, whenever such inapplicable codings are not isolated to a single group on the tree (W. Maddison, 1993). When the character is inapplicable to a taxon, in general it would be better to code it as having an extra state "not present". Although MacClade allows you to do that implicitly by entering the coding for a gap (by default, "-"), MacClade's algorithms treat the gap just as they do missing data. W. Maddison (1993) discusses some possible workarounds to this situation, but none are ideal. The missing data coding would be better restricted to those circumstances in which the organism is presumed to have one of the character states, but which one it has is simply not known because of incomplete information.

Inferring the tree

Now that we have introduced the vocabulary of trees and characters, we can discuss a central problem of phylogenetic biology — the reconstruction of evolutionary history. This involves first the reconstruction of the past genetic connections of organisms (the phylogenetic tree). Reconstructing the phylogenetic tree is no simple task, but at least we know where to start. The echoes of phylogenetic history arrive to us in the traits of the organisms it has produced.

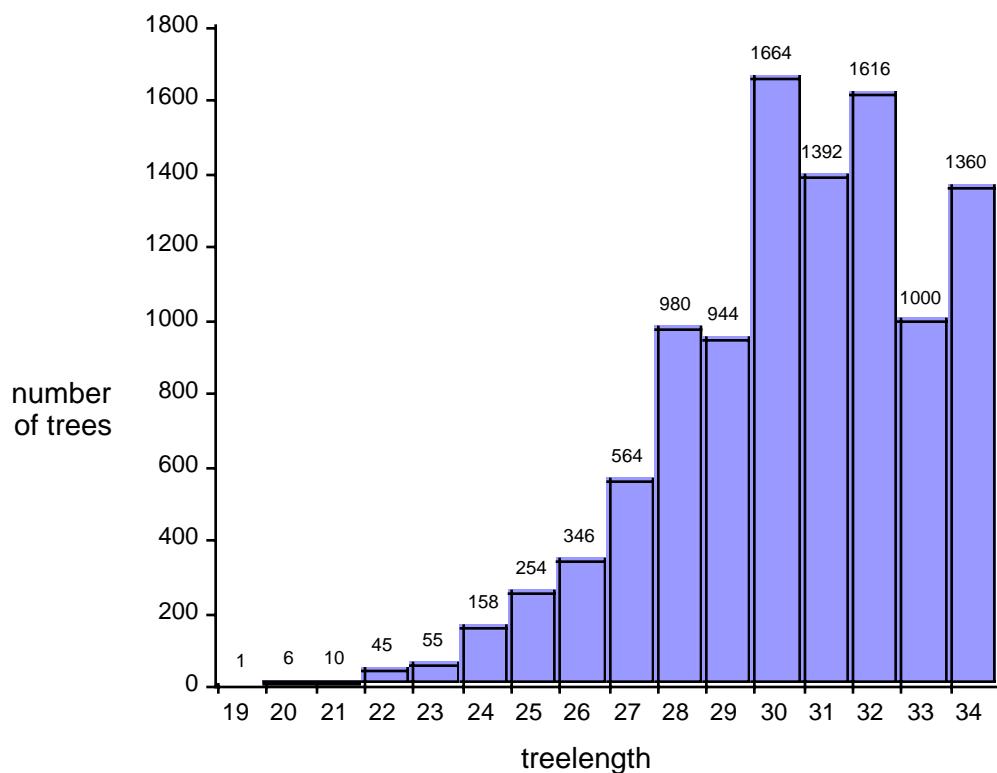
Similarities and differences among organisms have long been recognized as the primary evidence with which to reconstruct phylogeny, but exactly how this evidence is to be used has been the subject of considerable debate. The members of a clade share uniquely among themselves that branch of the tree that is their most recent common ancestor. Any new characteristics that evolved on that branch will mark the members of that clade, whereas species not part of the clade, that is, species not descended from this common ancestor, will lack these characteristics. By this logic Hennig (1966) argued that a clade can be recognized if its members share characteristics that are derived within the larger group of species being considered; that is, they have **synapomorphies**.

The clarity of this logic is unquestionable, but it does not fully prepare us for the real world. Character state distributions do not align themselves neatly into nested sets delimiting more and more restricted clades: Some characters disagree with others as to what monophyletic groups they suggest. Convergences and reversals serve to make members of different clades appear to be marked by the same apomorphy, or members of a clade sometimes to lack the clade's insignia. Some criterion by which to balance conflicting evidence from different characters is therefore needed.

Parsimony methods

If our goal is to find scientific hypotheses that can most simply explain the observed data, then we might apply the criterion of **parsimony** to the conflicting evidence, and choose the tree that requires the least amount of convergence and reversal among the characters (Edwards and Cavalli-Sforza, 1964; Camin and Sokal, 1965; Farris, 1970, 1983; Fitch, 1971; Sober, 1988). A measure of the amount of evolutionary change required by a tree, therefore indicating the amount of convergence and reversal, is the **treelength**. Treelength is calculated by the weighted sum of steps in the characters (["Basic Tree and Character Statistics" on page 373](#)). It therefore depends on whether we weight characters differentially, and on what models we have about character evolution (discussed below in the section on "Inferring the history of character change", and in [Chapter 4](#) and [Chapter 19](#)).

The treelength can be calculated for any tree. For example, the file "Vertebrates Small" contains a smaller version (only eight taxa) of the "Vertebrates" data matrix you used in the Tutorial. For eight taxa, there are 10,395 dichotomous, unrooted trees one could imagine; that is, 10,395 possible ways one could arrange the taxa into a tree. If one calculated the treelength of each of those trees, one would see that most trees have a treelength between 28 and 34, with some trees being **shorter**, one tree having a treelength of only 19. The distribution of these treelengths is plotted in the chart, below.



The distribution of treelengths for all 10,395 trees for the matrix "Vertebrates Small"

The tree of length 19 is called the **most parsimonious tree**, or **MPT**, for these data. For this matrix there is only a single MPT; for other matrices there can be multiple trees that are equally most parsimonious. Parsimony methods dictate that the tree or trees of choice are those that are most parsimonious.

The treelength is thus the score or measure of quality used to judge the trees. The formula used to calculate treelength ([page 374](#)) is the **objective function** used by parsimony methods. The **optimality criterion** used by parsimony methods further specifies that the trees of choice are those with the best score produced by the treelength objective function, where in the context of parsimony, "best" means "lowest".

A task of computer programs such as PAUP* (Swofford, 2000) and NONA (Goloboff, 1999) that implement parsimony methods is to calculate the length of various candidate trees as they attempt to find which of the conceivable trees is most parsimonious. The search for the most parsimonious trees is not always an easy one.

The simplest approach is to calculate the treelength of each of the possible trees. Once the treelengths of all of the trees are known, it will be easy to see which trees have the shortest length, and in this way the MPTs will be found. In this way one can be certain one has found the MPTs. However, because the number of possible trees becomes astronomically large as the number of taxa increases (Felsenstein, 1978a) this **exhaustive search** approach is feasible only with small matrices of few taxa, in part because it would take too long to calculate the treelength for all of the trees. PAUP* 4.0, for example, will conduct an exhaustive search only if there are 12 or fewer taxa.

Fortunately, there is a method guaranteed to find the MPTs for some data matrices that are too large to allow an exhaustive search. **Branch-and-bound** (Hendy and Penny, 1982; Swofford et al., 1996) is implemented in several computer programs; it allows one to find the MPTs with certainty for data matrices with as many as 25 or more taxa. The maximum number of taxa depends upon properties of the data.

However, for many matrices, even branch-and-bound is not feasible, and we are left with methods that search through possible trees in a much less thorough fashion. These methods, sometimes called **heuristic search** methods (Swofford et al., 1996; Swofford, 2000) are unfortunately not guaranteed to find the MPTs. Heuristic search methods come in many different varieties, and choosing an appropriate search strategy requires several considerations (D. Maddison, 1991a; Maddison et al., 1992).

One can conduct simple heuristic searches for the MPTs using MacClade, but its automatic search capabilities are not as well developed as those in other programs such as PAUP* (Swofford, 2000) and NONA (Goloboff, 1999). MacClade's role in inferring phylogenies is primarily as a tool to prepare your data and other parts of your files (such as constraint trees and other PAUP* commands; see [Chapter 26](#)), and to allow you to understand and process your results using tools such as the Compare 2 Tree Files chart ([page 405](#)).

Parsimony methods bear some relationship to Hennigian argumentation (e.g., Wiley, 1981:139). For instance, grouping by characteristics determined to be derived by outgroup analysis can be justified by parsimony considerations (Farris, 1982; Maddison et al., 1984). However, when convergence and reversal make evidence from different characters conflict, then the subtle balancing that can occur in parsimony decisions may not have any simple parallel in Hennigian argumentation.

There are several modifications one can make to treelength in order to incorporate various assumptions about character evolution. One can, for example, choose to adjust the treelength by weighting some characters more highly than others, or by weighting changes between some states more highly than changes between other states (as discussed later in this chapter).

Methods other than parsimony

Some methods of phylogenetic inference use scores other than treelength to judge the quality of a tree; that is, they use different objective functions and optimality criteria.

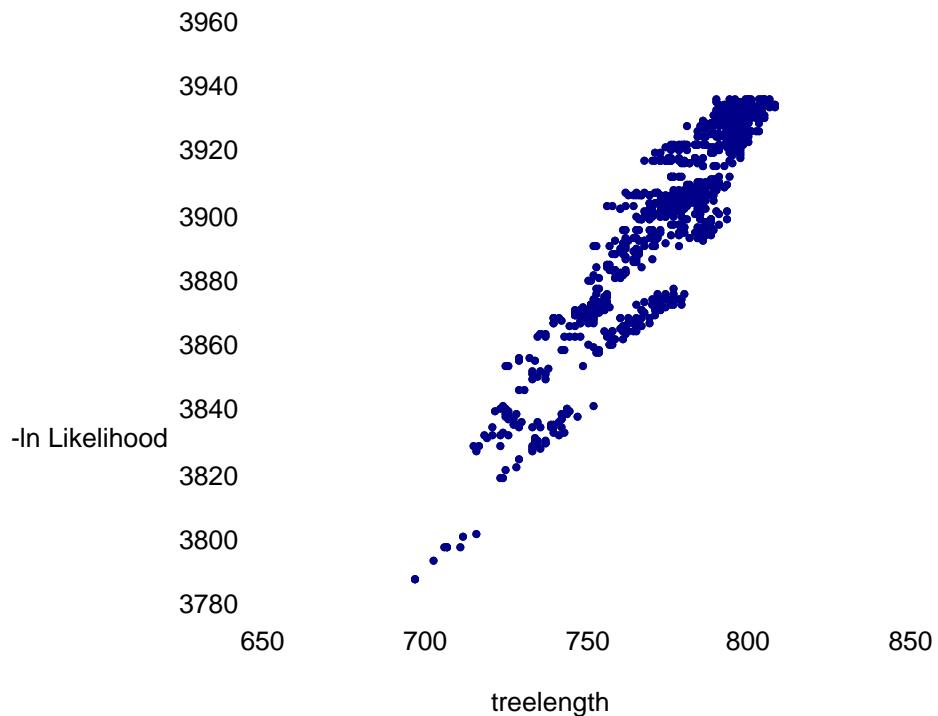
Rather than calculating the treelength of each tree considered, one might calculate the **likelihood** of a tree, which is the probability that the data matrix observed would have been generated by a specified evolutionary process on that tree (Edwards and Cavalli-Sforza, 1964; Edwards, 1970; Felsenstein, 1973, 1981b, 1992; Thompson, 1975, 1986; Barry and Hartigan, 1987; Kishino et al., 1990; Goldman, 1990; Weir, 1990; Yang, 1993, 1994a; Goldman and Yang, 1994; Yang et al., 1995a; Swofford et al., 1996; Huelsenbeck and Crandall, 1997). **Likelihood methods** use just such a measure of a tree, and choose that tree or trees with

the highest value of the likelihood, that is, for which the probability of generating the observed data is highest. For each tree considered, the branch lengths of the tree (in the sense of expected amount of change per character from the start to the end of the branch) are also estimated by choosing those branch lengths that maximize the likelihood of the tree. To date these methods have focused mostly on molecular and genetic data.

Likelihood methods rely on a model of the evolutionary process. The structure of the model can be simple or complex; it can specify the relative rates of change between the states of the characters, the nature of variation in evolutionary rate from one character to the next, and so on. The numerical values of various parameters in the model can be estimated using likelihood, by choosing those values that maximize the likelihood of a tree (Edwards and Cavalli-Sforza, 1964; Swofford et al., 1996).

As with parsimony methods, once the exact formula for calculating the likelihood of a tree is specified by choosing the assumptions, the next step is to search for the maximum likelihood tree or trees through an exhaustive, branch-and-bound, or heuristic search.

In general, one would expect that there will be a moderate correlation between the treelength and likelihood of a tree, such that trees of short treelength will be trees of high likelihood (or, equivalently, trees with low values of $-\ln \text{Likelihood}$), as shown in the following graph for the "Primate mtDNA Small" data matrix.



Treelength versus $-\ln \text{Likelihood}$ for all 945 trees for the "Primate mtDNA Small" data matrix.
 Treelength is calculated assuming all characters are unordered and with equal weighting;
 $-\ln \text{Likelihood}$ is calculated using PAUP*4 (Swofford, 2000) assuming a General Time
 Reversible rate matrix, each codon position evolving at a different rate, with base
 frequencies and parameter values estimated using likelihood on a most parsimonious tree.

In the above example, the tree of shortest treelength also happens to be the tree of lowest $-\ln \text{Likelihood}$ (and thus is the tree of highest likelihood), and although in this example parsimony and likelihood would

choose the same tree, in other examples they need not agree. Sometimes the maximum likelihood tree is not most parsimonious. It should be realized that the correlation between parsimony and likelihood analyses will depend on the assumptions used in each. With some assumptions, the correlation between parsimony and likelihood will be less than in the example above; with other assumptions, it could be more. Indeed, Tuffley and Steel (1997) have shown that a simple model of evolution yields a likelihood method that always chooses the exact same trees as simple unordered-state parsimony.

Most **distance methods** use another measure of quality of a tree, a measure that is derived in part from a matrix of pairwise distances between taxa. The exact nature of the measure varies from distance method to distance method. This is a heterogeneous assemblage of methods united by their use of pairwise distance matrices rather than by their goals or biological assumptions (Sokal and Sneath, 1963; Fitch and Margoliash, 1967; Cavalli-Sforza and Edwards, 1967; Sneath and Sokal, 1973; Bulmer, 1991; Saitou and Nei, 1987; Rzhetsky and Nei, 1992; Yang, 1994b; Steel, 1994; Lockhart et al., 1994; Lake, 1994; Swofford et al., 1996). Distance methods can use models of the same form as likelihood methods. Again, once the nature of the distance measure is specified, the trees with the optimum value of the measure are sought using the appropriate search method.

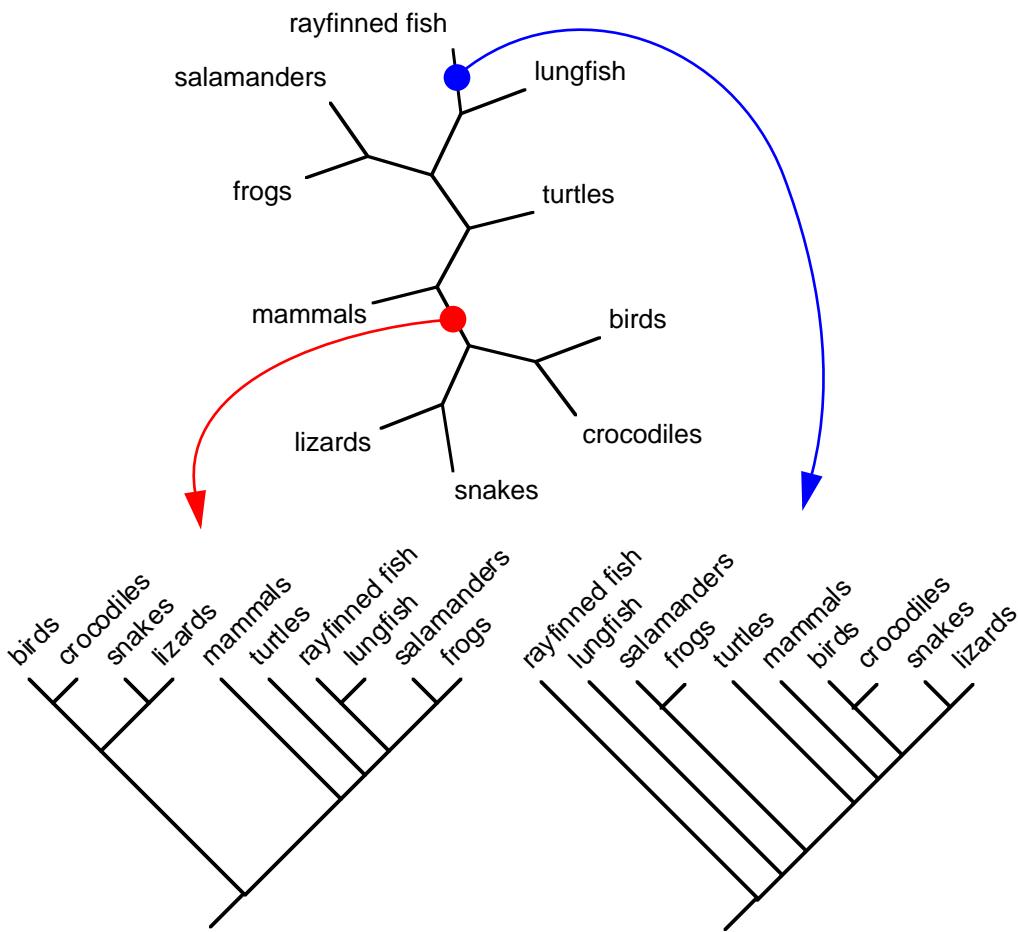
Other measures of quality of a tree include those used by invariant methods (Lake, 1987; Cavender and Felsenstein, 1987; Cavender, 1989; Jin and Nei, 1990; Sankoff, 1990; Sidow and Wilson, 1990, 1992; Navidi et al., 1991), compatibility analyses (Le Quesne, 1974; Estabrook et al., 1976; Meacham and Estabrook, 1985), and closest tree methods (Hendy, 1991), among others. These approaches are used less frequently than parsimony, likelihood, or distance methods.

One distance method, Neighbor Joining (Saitou and Nei, 1987), notable for its popularity, does not fit into the descriptions given above. It is an algorithmic method that produces a tree through a mechanical process of ever-finer resolution of an unresolved tree. It does not utilize a measure of quality (objective function) by which trees can be compared one to another.

With many different methods of phylogenetic inference, each with its own variants, the choice among methods faced by practicing systematist can be daunting. Considerations relevant to this choice include the method's philosophical appropriateness, its practicality (has a variant of a method been built for the type of data at hand? can the analysis be conducted in a reasonable length of time?), its consistency (is the method more likely to yield the correct phylogeny if more data are added?), its power (how much data are needed for the method to perform well?; this is sometimes called efficiency), and its robustness (under how broad an array of conditions will the method perform well?). For more information relevant to a choice among methods, the reader is referred to the reviews by Felsenstein (1982, 1988a), Penny et al. (1992), Hillis et al. (1994), Swofford et al. (1996), and Steel and Penny (2000).

Rooting the tree

For most of the measures of tree quality used by parsimony, likelihood, or other methods, an entire tree can be rerooted on any of its branches, and each of these rerooted trees will have the same score. For example, with the "Vertebrates" example file, if one rooted the tree next to ray-finned fish, on the blue dot shown in the diagram below, the treelength would be 19; if one roots it at the red dot, on the branch separating birds, crocodiles, snakes, lizards, from the rest of the tree, the treelength is still 19. Any of the 17 possible rootings would yield a tree of length 19.



Ideally, our methods of tree inference would allow us to choose which of these branches is the ancestral branch — that is, which branch is the root of the tree. In the absence of an inferred root, we are left with a collection of candidate rooted trees, which in sum form a kind of consensus hypothesis called an unrooted tree.

To choose the root of the entire tree or part of it, we need additional data or assumptions. These can come in several forms:

1. Assumptions about the ancestral state in one or more characters. If one has evidence that a particular state in a character is the ancestral state for the taxa contained in the tree, then it will be more parsimonious to root the tree among those branches for which that state is parsimoniously reconstructed. In making the claim about ancestral state, one is presuming that at least one of the changes on the tree will be from that state to another. This is a weak statement about the polarity of character change, and has been called **basal polarity** (Maddison and Maddison, 1992).
2. Assumptions of direction of evolution throughout the tree. One can invoke a stronger claim about the polarity of character state change, specifying that throughout the entire tree all changes for a character are in a particular direction, or that the changes are more likely in one direction than another. This specification of **global polarity** can be invoked by presuming a character is irreversible, or by presuming an asymmetrical cost matrix (cost matrices are introduced on [page 56](#)). These assumptions make some tree rootings more parsimonious than others. It is perhaps obvious that assumptions such as character irreversibility can help root the tree, but milder assumptions

about global polarity, such as those embodied in asymmetrical step matrices, can also have the power to choose the root of the tree (Maddison and Maddison, 1987; Wheeler, 1990).

3. If one incorporates into a tree inference not only the study groups but also any outgroups, then the root of the study group tree will be the branch that connects to the outgroups (Farris, 1972). Such an analysis will require assuming some phylogenetic structure, namely what groups might be considered outgroups and what groups comprise the study group. At no point in the analysis would one actually make an explicit assumption about ancestral states. In some circumstances including outgroups as extra taxa is equivalent to use of assumptions about character polarity, if the outgroup data are condensed into a claim about basal polarity (Maddison et al., 1984). However, in other circumstances, particularly when outgroup relationships are uncertain, it is better to avoid specifying character polarity and instead include all the outgroup taxa along with the study group during tree inference (Farris, 1972; Maddison et al., 1984).
4. By analyzing the composite phylogeny of duplicated genes, the point at which subtrees of separate gene copies attach to one another can indicate the root of the species tree (Iwabe et al., 1989). One can extend this method, and choose the root of a species tree to minimize the number of duplications and extinctions of genes required to fit a gene tree within the species tree (Maddison and Maddison, 1992:53). That is, for each candidate species tree, one can fit the contained gene tree so as to minimize the required number of duplications and extinctions of genes. One can then choose the rooted species tree with the minimum number of duplications and extinctions on the inferred gene tree.

The last two procedures can be used with any method of phylogenetic inference.

How can you specify polarity or outgroups in MacClade? MacClade has no facility to specify directly basal polarity by designating which states are ancestral, as with PAUP*'s ANCSTATES command. Therefore, to use an assumption about ancestral states, you must use outgroups, by entering a hypothetical taxon that has the proposed ancestral states, and maintaining it near the base of the tree as an outgroup. You can specify global polarity by setting a character's type to irreversible or an asymmetrical, user-defined type. Finally, you can incorporate observed outgroups directly into the analysis and maintain them near the base of the tree. MacClade has no facility to enforce the designation of some taxa as outgroups. Therefore, if some of your taxa are outgroups, it is your responsibility to maintain them apart from the ingroup and impose any phylogenetic structure within the outgroups you wish to assume.

Branch lengths

Although the first concern in reconstructing a phylogenetic tree is to determine the order of branching events that gave rise to the taxa, a second concern might be to reconstruct the lengths of the branches of the tree. "Branch length" is a term whose meaning varies in the literature. It can refer to:

1. The length in time between the node just ancestral to the branch and the node terminating the branch.
2. The expected amount of character change on the branch per character.
3. The actual number of evolutionary changes along the length of the branch.
4. The actual number of differences between the characters in the organisms at the start of the branch and the end of the branch. (This can differ from the previous if there is more than one change on the branch — evolution from A to G to A along a branch implies two actual changes but no net difference from start to end of branch.)
5. The reconstructed number of evolutionary changes over all observed characters along a branch.

To know branch lengths in the first sense would be useful to evolutionary biology, but these time lengths are not easy to come by. Paleontological data can give us minimum ages of nodes. Molecular data can date nodes but at the cost of assumptions about the rates of molecular evolution (Thorne et al., 1998; Huelsenbeck et al., 2000).

In maximum likelihood estimation of trees, branch lengths in the second sense often come included as part of the estimated tree. In parsimony analyses, branch lengths tend to be derived secondarily, after trees are reconstructed. One way to estimate 2 to 4 would be to reconstruct ancestral states and use a reconstruction of character evolution to calculate probabilities, changes or differences. However, parsimony reconstructions usually underestimate the actual number of changes along the branch (Felsenstein, 1985a; Saitou, 1989). Parsimony reconstructs parallel changes on sister branches, or multiple changes along a single branch, as a single change. Various algorithms have been devised to estimate branch lengths under the second interpretation listed above, and these generally assume a stochastic model of evolution that allows one to estimate and incorporate the number of hidden changes into the estimate of branch length (Fitch and Bruschi, 1987; Hendy and Penny, 1989; Fitch and Bientema, 1990; Olsen, 1991; Tajima, 1992).

In parsimony-based computer programs, the branch length usually refers to the number (or amount, if weighted) of reconstructed changes on the branch. The number of reconstructed changes on a branch is sometimes taken to be an indication of support for the clade above it, with longer branches taken to provide greater evidence for the monophyly of the members of the clade. However, this interpretation is problematic, in part because the branch length is usually ambiguous as a result of equally parsimonious reconstructions of change, and in part because this ignores interactions between characters (see the sections "[Polymorphisms and missing data](#)" on page 43 and "[Uncertainty in the reconstruction](#)" on page 100).

Using phylogenetic trees to interpret evolution

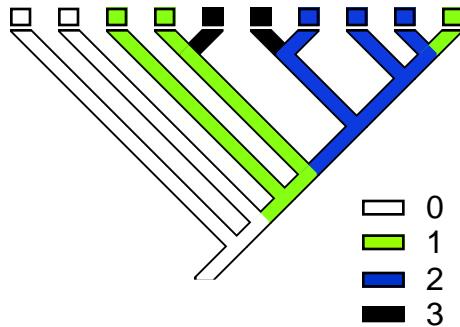
In this chapter we have focused up to this point on reconstructing trees. Although MacClade can assist in tree reconstruction, most of its analytical tools are designed to interpret evolution using trees already reconstructed. The strong growth of phylogenetic biology in the last two decades has been fueled by a recognition of the importance of a phylogenetic perspective in interpreting evolution.

In the remaining sections of this chapter we will deal with some issues relating to evolutionary interpretations using trees, especially concerning character evolution. We will not attempt to present an exhaustive account of the uses of phylogenetic trees. An overview of this topic is given in the previous version of MacClade (Maddison and Maddison, 1992: Chapter 3), and more recent reviews have been given by D. Maddison (1994), W. Maddison (1996), Harvey et al. (1995), Huelsenbeck and Crandall (1997), Huelsenbeck and Rannala (1997), and Pagel (1997, 1999a).

Character evolution: History versus models

It is along the branches of phylogenetic trees that the traits of organisms have evolved. In studying phylogenetic character change, we can seek either a narrative description of the history of change, or a model of the process of change. The discussion in the next sections covers these two, distinct aspects of our hypotheses of character change, and is based upon D. Maddison (1994).

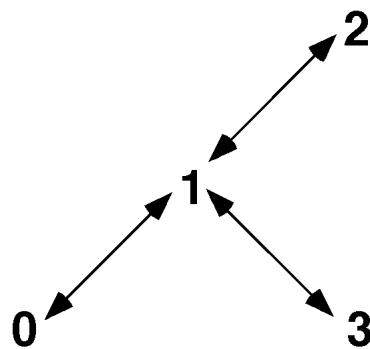
Imagine we have inferred a phylogeny for our study group. We might wish to know how a particular trait of interest evolved along the lineages of this phylogeny. We might propose a hypothesis of the pathways of change of the characteristic, as shown below.



Hypothesis about the evolution of a particular character on a phylogeny. States of the character are indicated by shadings.

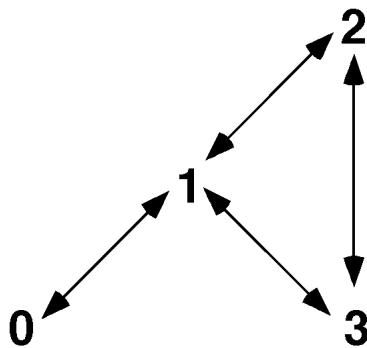
This hypothesis states that the ancestral lineage of the organisms had state 0 (indicated by white), with a later lineage evolving to state 1 (green), and then to state 2 (blue), and so on. This is a hypothesis about the **history of character change**: that is, what actually happened, lineage by lineage, during the course of evolution. It is a hypothesis about the history of the pathways taken by characters from one state to another through the course of evolution. In later chapters we will speak of "**reconstructions** of ancestral states", or "**tracings** of character evolution", or "a mapping of the character on the tree". These phrases are interchangeable; they amount to hypotheses of the history of character change.

It is important to distinguish between what happened and our models as to how evolution in general happens. For instance, we might have reason to believe that some changes between states are possible whereas others are not. This would be a **possible-pathways model** (D. Maddison, 1994). For example, in the possible-pathways model pictured below, a direct change from 0 to 1 is considered possible, but a direct change from 0 to 3 is not.



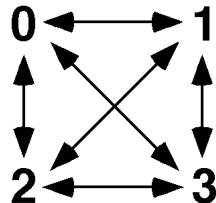
A possible-pathways model
(character state tree)

A possible-pathways model such as the one pictured above, without closed circuits, can also be called a **character state tree**. However, if circuits are present, as in the figure below, then the more general term of **character state graph** is appropriate.



Another possible-pathways model
(character state graph)

The most extreme character state graph in terms of the pathways allowed connects every state directly to every other state:



Such a graph depicts the model underlying unordered characters, for which a change from any state to any other state can occur in a single step.

Although character state tree diagrams look like little phylogenetic trees linking character states, they are not, and they differ in the fundamental sense that phylogenetic trees purport to show what actually happened in evolution, whereas character state trees show what can in general happen in evolution. There has been some confusion on this point. The term "transformation series" has been applied to both the actual history of character change (e.g., Liebherr and Hajeck, 1990) and possible-pathways models (e.g., Mickevich and Weller 1990). There is also variation in the meaning of "character phylogeny", although the term is not as widely used. Hennig's use (1966:95) appears to correspond to the actual history of character change. Mickevich (1982:461) equates the term to a possible-pathways model; Mickevich and Weller (1990) use it as synonymous with the history of character change.

One might also imagine a model that specifies not just the changes that are possible, but the relative rates or probabilities of such changes. For example, imagine that one presumes that from the start to the end of each branch on the tree there is a particular probability that a 0 will evolve into a 1, or that a 2 will remain a 2, and so on. The **substitution probability matrix** below, for example, specifies that a branch beginning with state 0 has a probability of 0.90 of ending at state 0, 0.04 of ending at state 1, 0.02 of ending at state 2, and 0.04 of ending at state 3.

To:

| | 0 | 1 | 2 | 3 |
|---------|------|------|------|------|
| From: 0 | 0.90 | 0.04 | 0.02 | 0.04 |
| 1 | 0.08 | 0.80 | 0.10 | 0.02 |
| 2 | 0.04 | 0.06 | 0.85 | 0.05 |
| 3 | 0.20 | 0.12 | 0.08 | 0.60 |

Model specifying probabilities
of change between states between
the start and end of each branch

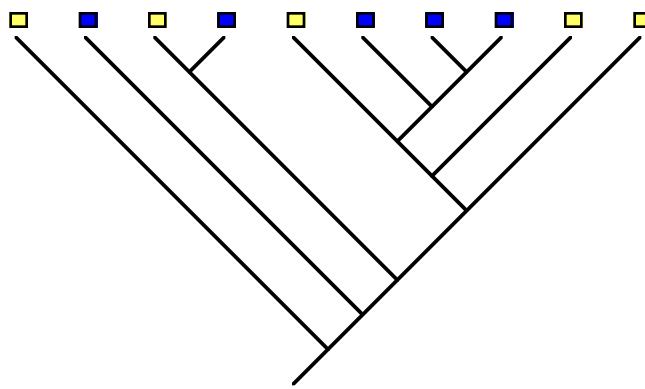
One could presume that this probabilistic model of change applies to all branches in the tree, or that it varies from branch to branch, with some branches having higher probabilities of change than other branches. One natural way to allow some variation from branch to branch across the tree is to separate the probabilistic model of change into two components: a matrix specifying the rates of change from one state to another across an infinitesimally small portion of a branch, and the length of each branch. Along any particular branch, this model would specify a substitution probability matrix, which could be calculated by combining the rate matrix and the length of the branch. The longer the branch, the more the rate matrix can "act", and the higher the probability of change from the start to the end of the branch. The rate matrix might be presumed constant over the entire tree, with differences in probabilities of change among any branches created by the variation in branch length. This is part of the model used in most implementations of maximum likelihood.

Hypotheses about what happened and what in general might happen, though fundamentally distinct, have an intimate relationship: in order to reconstruct the history of what happened, we need to assume something about what in general could happen, and in order to formulate theories about what in general happens in the evolutionary process, we may need to know something about what actually happened in evolution. Obviously, one needs to step gingerly to avoid circularity, but it can be done, and we will now consider further the principles of inferring the history and the processes.

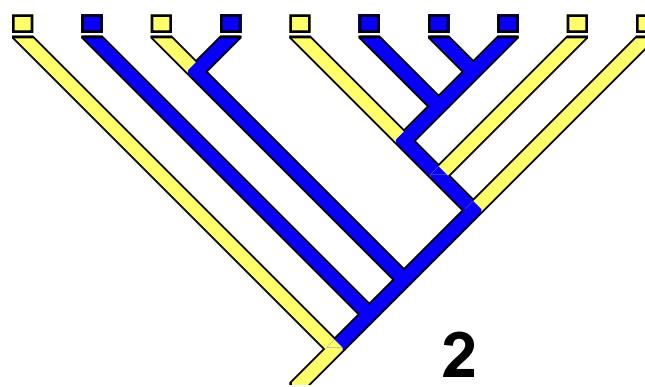
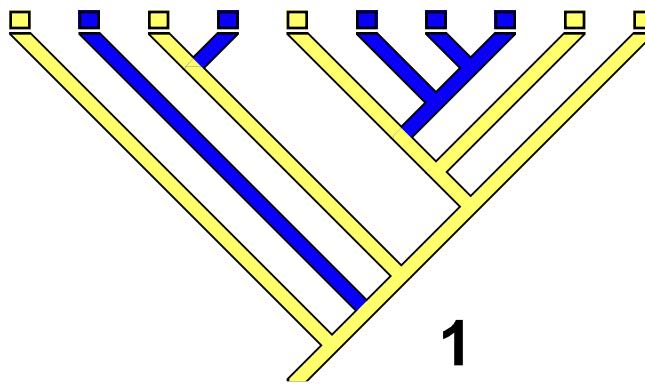
Inferring the history of character change

How to reconstruct the history of character change? We will presume that the phylogenetic tree is known, and that we have a character whose states are known in the taxa sampled. Our goal is to reconstruct the history of character changes along the branches of the phylogeny.

For a study group, with known phylogeny, there are numerous ways a particular character could have evolved resulting in the states observed in the terminal taxa. For example, for the following phylogeny,



two possible historical reconstructions that would lead to the observed states are:



For this example, there are nine internal nodes, each of which could have two states; there are thus $2^9 = 512$ possible reconstructions. Assuming our goal is to reconstruct the history of character changes during evolution, how do we choose among these hundreds of possibilities?

Currently, the most commonly used numerical method to reconstruct ancestral states is based on the criterion of parsimony. In its most basic form, **parsimony** chooses the historical reconstruction with the minimum number of evolutionary changes (Farris 1970, 1983; see also [Chapter 4](#)). Between the two

reconstructions shown in the example above, we would thus choose reconstruction 1, as it entails 3 changes (three changes from white to black), over reconstruction 2, as it involves 5 changes (one white to black change and four from black to white). One could examine all 512 possible reconstructions, in the process discover which of these have the fewest number of changes, and chose those **most parsimonious reconstructions** (or **MPRs**). (In fact, one needn't examine all the possible reconstructions, as there are methods available to find the MPRs without examining all reconstructions, as discussed in the chapter "[Reconstructing Character Evolution Using Parsimony](#)".)

Parsimony methods can incorporate some notions about evolutionary process even if they are not explicitly statistical. For example, you might assume that character evolution follows a character state tree as discussed above, or you might assume that Dollo's law (Farris, 1977) applies to a character. These assumptions place constraints or weights on the reconstructed character changes, and therefore alter which reconstruction is judged most parsimonious. The weight placed upon a particular sort of change is called its **cost**.

When changes are weighted in this way, the quantity that parsimony calculations attempt to minimize in judging reconstructions is the summed cost of all the changes. This summed cost is referred to as the number of **steps** in a character. (When weighted and summed over all characters it is the **treelength**.) Even though the term "step" would seem to connote a single discrete evolutionary change, in MacClade and PAUP* "step" is used in a more general way as a unit of cost. A single evolutionary change might be measured as costing 1 step, or 2 steps, or even 1.31 steps (because MacClade allows continuous-valued costs). Further details concerning changes, steps, and lengths are given in [Chapter 4](#) and [Chapter 19](#).

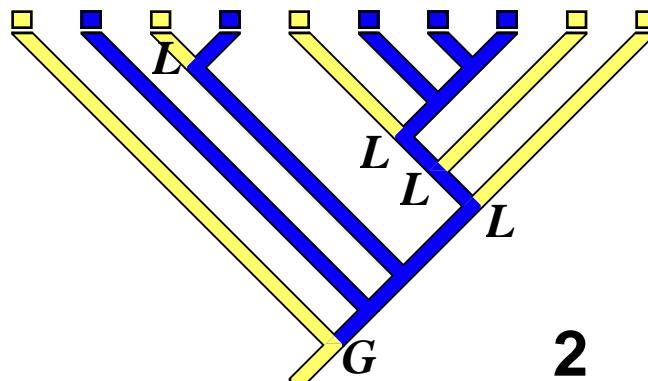
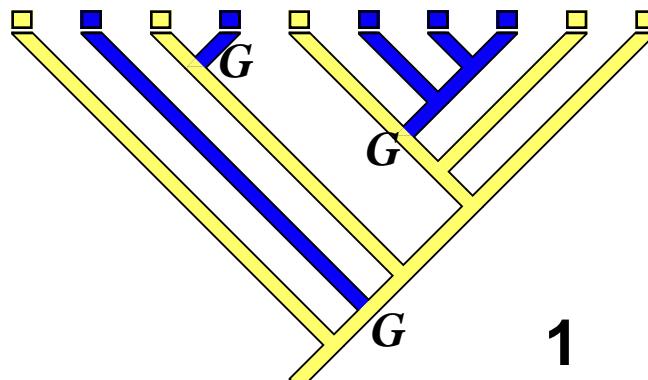
Among the key assumptions used in a parsimony analysis therefore are the assignments of relative weights or costs to each type of change. The costs assigned to each type of change can be summarized in a **cost or step matrix** (for further details see [Chapter 4](#)). Although the term "cost matrix" (D. Maddison, 1990) is more general and perhaps more appropriate, we will use the term "step matrix" following recent use (Swofford, 1991; Swofford et al., 1996). If we assign a cost of **G** to each gain of blue, and a cost of **L** to each loss of blue, we would have the step matrix shown below.

| | | To: | |
|-------|--------|--------|------|
| | | Yellow | Blue |
| From: | Yellow | 0 | G |
| | Blue | L | 0 |

A cost or step matrix

Under this step matrix, reconstruction 1 entails a total cost of $3G$, and reconstruction 2 entails a cost of $4L +$

G:



The upper reconstruction would be considered more parsimonious than the lower reconstruction if $3G$ was less than $4L + G$. Thus, if L and G were both 1, the upper reconstruction would be more parsimonious. If G were 2 and L were 1, the lower would be more parsimonious. Of course, there might be another reconstruction that is even more parsimonious.

Even after one has fully specified the assumptions behind a parsimony analysis, there remains the issue of how to find the reconstruction that is most parsimonious. Various algorithms have been devised to find the most parsimonious reconstruction; these are used by MacClade and are described in detail in the chapter ["Reconstructing Character Evolution Using Parsimony" on page 66](#). Often there will be more than one most-parsimonious reconstruction — in fact, there can be many millions of reconstructions of equally low cost.

The accuracy of the ancestral states reconstructed by parsimony depends on the rates of evolution and the lengths of the lineages along which change might happen (D. Maddison, 1994; Frumhoff and Reeve, 1994; Hillis et al., 1994; W. Maddison, 1995; Schultz et al., 1996; Martins, 1999). Different authors take a more or less optimistic view of the performance of these methods. Given the difficulty of the estimation problem (Maddison and Maddison, 1992; Omland, 1999), biologists should always take into account the possibility of errors in their reconstructions of ancestral states.

In the last several years there has been increasing attention to alternative methods of reconstructing ancestral states using **maximum likelihood** and related methods (W. Maddison, 1991; Maddison and Maddison, 1992: 60; Schluter, 1995; Yang et al., 1995b; Koshi and Goldstein, 1996; Schluter et al., 1997; Mooers and Schluter, 1999; Pagel, 1999b; Schultz and Churchill, 1999). These statistical methods explicitly assume a probabilistic model of character evolution in order to make historical inferences. They vary not only in the model of evolution used, but whether parameters of the model of character change are estimated simultaneous with the inference of ancestral state and how they are estimated. Both this version and the previous version (Maddison and Maddison, 1992) of MacClade can do one, limited type of likelihood-related ancestral state reconstruction, when a step matrix is used that indicates negative logarithms of probabilities of change (see below).

It is clear that much work remains to determine both the best methods to reconstruct ancestral states and their accuracies. Recent reviews of ancestral state reconstruction are given by D. Maddison (1994), Cunningham et al. (1998), Omland (1999), and Chang and Donoghue (2000).

What are the steps in a step matrix?

We have just discussed parsimony methods that find reconstructions of character change that minimize the total cost of character changes, where costs are measured implicitly or explicitly according to a cost or step matrix. The step matrix is therefore an assumption used in the reconstruction. We will here consider the sources from which this step matrix might be derived, in order to emphasize that it is based on a model of how evolution happens.

In the discussion above we described character state trees and character state graphs as possible-pathways models of character evolution. Assuming that our character evolves according to a possible-pathways model, then the cost of presuming a state i to a state j change could be the number of evolutionary steps implied on the model. For example, if one assumed that the states were linearly ordered, then the number of steps between states 0 and 1 is 1, between states 1 and 3 is 2, and so on, implying the cost or step matrix shown below.

0 ←→ 1 ←→ 2 ←→ 3

Linearly ordered states

To:

| | 0 | 1 | 2 | 3 |
|-------|---|---|---|---|
| From: | 0 | 0 | 1 | 2 |
| | 1 | 1 | 0 | 1 |
| | 2 | 2 | 1 | 0 |
| | 3 | 3 | 2 | 1 |

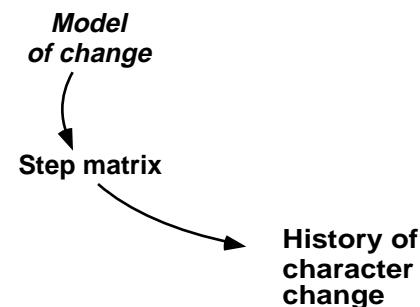
Step matrix derived
assuming linearly ordered states

The step matrix might be derived instead from a probability model of the relative probabilities of change between states along each branch, as described above. To derive a step matrix from the probability model one could suppose that the more likely a change, the less it costs. One possible conversion function (Felsenstein, 1981a; W. Wheeler, 1990) is:

$$\text{Cost} = - \text{natural logarithm of the probability}$$

Indeed, if a step matrix is used whose costs are derived from a substitution probability matrix using the above logarithmic formula, then a most parsimonious reconstruction of ancestral states will be more than a parsimony estimate. It will also be a likelihood-related probability estimate, because the ancestral state reconstruction that minimizes the sum of negative logarithms of probabilities is also maximizing the product of probabilities of the implied change (or lack thereof) along the branches (D. Maddison, 1990; see also W. Maddison, 1991). The step matrix, however, might look rather unfamiliar to users of parsimony, because it will have the diagonal elements nonzero. It will cost something not to change! (MacClade does allow diagonal elements to be nonzero, and therefore can make probabilistic ancestral state reconstructions in addition to more typical parsimony reconstructions.) The main limitation on MacClade's use of this technique is that all branches must be treated as having the same probabilities (regardless of branch length.)

Whether from a possible-pathways model or a probabilistic model, we could view the step matrix, used for inference of the history of character change, as emanating from some model of change. Thus some model of how evolution in general happens is used in reconstructing the history of character changes. This is true even for phylogenetic analyses that assume characters are unordered. For an unordered character, the model of evolution says, in a sense, that anything can happen.



Analyzing characters on their own trees

An often-heard prescription is that characters that are analyzed in the light of a reconstructed phylogeny should not have been used to reconstruct the phylogeny (Coddington, 1988; Brooks and McLennan, 1991). The phylogeny, it is said, should have been derived independently from the characters later analyzed. This prescription is not followed universally, however, for many authors place tick marks on tree branches indicating interpreted evolutionary changes in characters that were used to reconstruct the tree.

Although the problem is sometimes characterized as one of circularity, it is better characterized as one of bias (W. Maddison, 1990; Swofford and Maddison, 1992; D. Maddison, 1994; de Queiroz, 1996). As noted by W. Maddison (1990), the inclusion of characters in reconstructing a phylogenetic tree will tend to reduce their own inferred number of convergences and reversals, at least if parsimony was used in reconstructing the tree. This biasing effect may in fact not be serious, and it may be conservative with respect to the evolutionary hypothesis investigated. de Queiroz (1996) gives a thorough review of these issues.

Of course, the better supported is the phylogeny by independent characters, the more confidence we will have in our character analysis. A phylogeny that is well-supported by independent characters would be preferred. However, Swofford and Maddison (1992) note that it is important to consider characters in the reconstruction of the phylogeny if they have a substantial contribution to make, even if those characters are later analyzed using the phylogeny. If you are reluctant to include characters you will later analyze, you should at least perform a sensitivity analysis to see what influence these characters would have had on the phylogeny. If addition of these characters significantly changes the phylogeny reconstructed, and thus casts doubt on the phylogeny reconstructed without them, then that phylogeny is probably not strongly enough supported to be enforced as a background assumption in the character analysis. (See the discussion on acceptable assumptions later in this chapter.)

Inferring models of character change using phylogenies

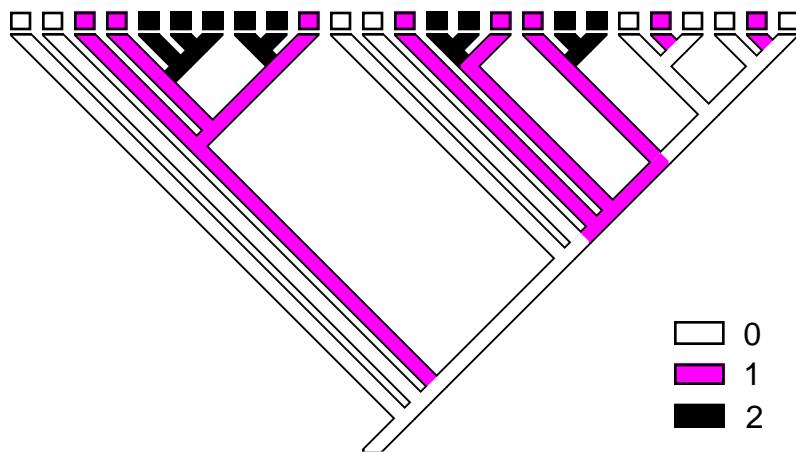
In the previous section on reconstructing the history of character changes, it was noted that the assumed matrix of costs used by parsimony implies a model of character evolution. What if your focus is not the history but rather the model of character evolution? In this section we will turn the previous discussion upside down by considering how models of character change may be derived using phylogenetic trees.

Models of the process of character change could be obtained from outside of systematics from such sources as developmental biology or molecular biology. For instance, developmental studies may indicate the nature of gene changes required to change one structure into another, and thereby suggest the relative probabilities of change. Alternatively, we may seek to use phylogenies to uncover the patterns of character change that would then allow us to formulate models of change. MacClade can help with phylogenetic inference of models of character change.

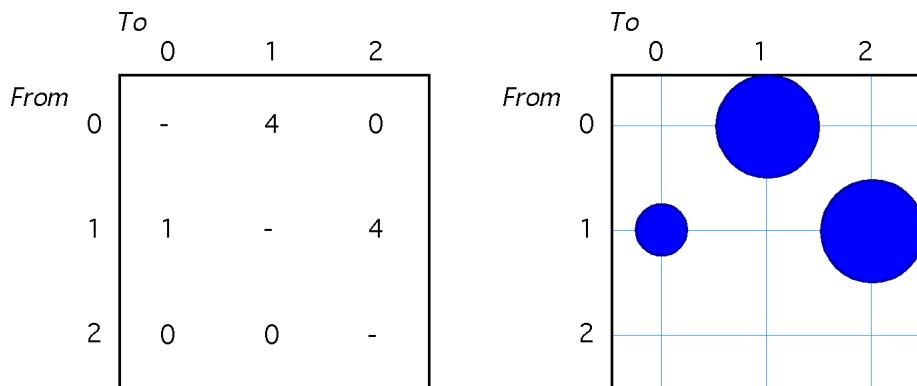
Phylogenetic methods use the distribution of character states on reconstructed phylogenies to infer the parameters of models of character change. Both parsimony-based and likelihood methods are available.

Parsimony methods for inferring character change models

To infer relative probabilities of change using a parsimony-based technique, several methods are possible. The simplest, used by MacClade's **State Changes & Stasis** chart (see [Chapter 20](#)), involves first reconstructing the history of character change and then extracting frequencies of each type of change. For example, the following inferred history of character change shows four 0→1 changes, one 1→0 change, and four 1→2 changes:

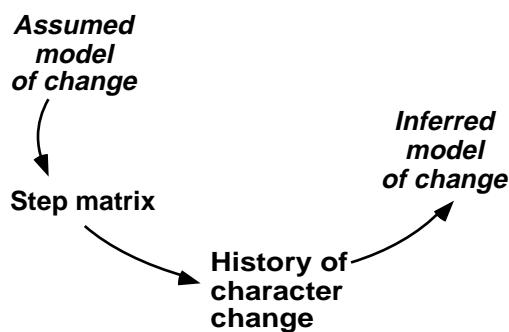


One could represent these relative frequencies of changes using the following table (on the left) or a "bubble diagram" (on the right):



Here, the area of each circle represents the relative frequency of that kind of change. Thus, $0 \rightarrow 1$ and $1 \rightarrow 2$ changes are more frequent than $1 \rightarrow 0$ changes; other changes were inferred not to have taken place.

One may wish to interpret this bubble diagram as an estimate of the parameters of a probabilistic model of the process of character change. The reconstructed frequencies of changes would be interpreted as estimating the actual probabilities of changes. There are several difficulties with this interpretation; we will mention only three here. First, the sample size is rather low (only 9 changes in all were inferred). In order to increase the sample size, one would have to examine changes on a much larger tree (in which case one would have to assume constancy of process across more lineages) or over multiple serially homologous characters, such as along a DNA sequence (in which case one would have to assume constancy of process across characters). Second, if the history of character change was reconstructed using the principle of parsimony, as it is in MacClade, then the bubble diagram may distort the probabilities because it seeks the reconstruction minimizing change. This distortion would be especially noticeable if one were using parsimony reconstructions to try to estimate the relative frequencies of branches with change as opposed to stasis — branches on which no change is reconstructed. An example of parsimony's distortion is given in the section "[Reconstructions used in State Changes & Stasis charts](#)" on page 397. Third, this estimation method contains an inherent bias arising from the way parsimony counts costs. This bias, which we will call **step-matrix bias**, was recognized by Mickevich (1982; Mickevich and Weller 1990).



The parsimony method used in the reconstruction method depends on an assumption of a step matrix, and that step matrix itself is derived from an assumed model of change. Clearly the assumed model of change can bias the reconstructed frequencies of change, and thus the inferred model of change.

(This method need not involve circular logic if, for instance the input model of change supplies a general constraint on the inference, whereas the output model of change is a more detailed hypothesis about process; see "[What assumptions are acceptable?](#)" on page 64.)

In an attempt to avoid this bias, Mickevich (1981, 1982), as part of her Transformation Series Analysis, introduced a numerical method that did not begin with a step matrix or character state tree, but rather derived character state trees during the course of the analysis from the distribution of character states on a tree. However, as outlined in Maddison and Maddison (1992) and D. Maddison (1994), this method is not well defined or justified.

Without clearly justified procedures that avoid an initial specification of step matrices or similar assumptions, it would appear that we are forced to use such assumptions. Despite the fact that step-matrix bias is a problem, it need not be devastating. Although the nature of the step matrix may obscure the true nature of change between character states, it does not necessarily hide it completely: some aspects of the relative frequencies of change can be detected despite the obfuscation created by step-matrix bias (D. Maddison, 1994). However, no fully satisfactory parsimony-based solution has been proposed. For this reason, we must treat parsimony estimation of process models with caution. We cannot fully escape our assumptions, but neither are we so fully bound by them that we can learn nothing new.

Likelihood methods for inferring character change models

Maximum likelihood approaches to estimating parameters of character change have been described by, among others, Barry and Hartigan (1987), Ritland and Clegg (1987), Goldman (1993a, 1993b), Sanderson (1993), Pagel (1994), Yang (1994b), Swofford et al. (1996), Lio and Goldman (1998) and Goldman and Whelan (2000). Derivation of stochastic models of change has been more of concern with molecular sequence data (previous citations and Gojobori et al., 1982; Lanave, et al., 1984; Fitch, 1986; Nei, 1987; Rempe, 1988) than with morphological data (e.g., Janson, 1992), in part because it seems more reasonable to presume constancy of models across nucleotides or amino acids than across numerous morphological characters.

In contrast to parsimony methods, likelihood methods provide estimates of the character change model without ever having reconstructed the history of character change (see next section).

Should we rely on historical reconstructions?

Methods to estimate parameters of character evolution can be grouped into two categories. Some, like the parsimony methods described above, reconstruct character changes on a phylogenetic tree, then use these changes to estimate parameters, such as the probability of different sorts of changes, whether two characters are correlated in their changes, and so on. Ridley (1983), Huey and Bennett (1987), W. Maddison (1990), Pagel and Harvey (1989), and Sanderson (1991) all describe methods that begin by making a reconstruction of the history of character change, and then proceed to use that reconstruction to make an inference about evolutionary process. Other methods avoid making a reconstruction of changes, and instead directly test for parameters using the observed distribution of character states, the form of the tree, and some stochastic model of evolution to relate them. Felsenstein (1985a), Barry and Hartigan (1987), Ritland and Clegg (1987), Golding and Felsenstein (1990), and Pagel (1994) all have described methods that estimate values of parameters of character evolution without at any point actually reconstructing the history of changes. For instance, a stochastic model can yield a likelihood, which is the probability of obtaining the observed data given the particular tree and parameters of character evolution. By seeing which parameter values yield the highest likelihoods, possibly comparing them with likelihood ratio tests, estimates of those parameters can be obtained. At no point is a reconstruction made of character change. In fact, these procedures legitimately sum probabilities over all possible scenarios of character change (even the least parsimonious). Likelihood methods are not the only ones to avoid a reconstruction of ancestral states. For example, pairwise comparisons (Felsenstein, 1985; Read and Nee, 1995; W. Maddison, 2000) can be used to test independence of character distributions without reference to ancestral states.

The problems of relying on an historical reconstruction of character change are several (Maddison and Maddison, 1992; Pagel, 1994). First, one tends to forget the error and ambiguity in the reconstruction of ancestral states (see [Chapter 4](#); Felsenstein, 1985a; W. Maddison, 1990; Swofford and Maddison, 1992; W.

Maddison, 1995; Cunningham et al., 1998). Second, one needs to be careful that the methods that reconstructed the ancestral states use assumptions consistent with the assumptions used in any subsequent estimation procedure based on the reconstructed states. One would not want to build a statistical test based on an elaborate stochastic model, only to apply it to ancestral states reconstructed using a method inconsistent with that model. Because of these problems, the direct methods that avoid the historical reconstruction have advantages.

This does not mean, however, that we should abandon making historical reconstructions of character change (and perhaps therefore abandon much of MacClade's use!). First, the history of evolution might be considered just as legitimate a subject of study as are the processes of evolution. Second, in many studies it is important to estimate the sequence of evolutionary events (Donoghue, 1989). As long as the possible errors are considered, the reconstruction can be important in understanding evolutionary processes. It serves at least the heuristic purpose of suggesting patterns, even if those patterns are later confirmed statistically by methods that avoid historical reconstructions. Third, direct methods that avoid historical reconstructions tend to require detailed stochastic models that may not be appropriate. Fourth, for many estimations we may want to make, direct methods are simply not yet invented, and we may find the indirect, historically based methods easier to concoct. As Felsenstein (1985a) has noted, it is probably better to use phylogeny crudely than to ignore it entirely. Fifth, in some circumstances simulation studies have shown that reconstructions of character change can be used in statistical methods without too much danger. W. Maddison (1990) and Martins and Garland (1991) both used simulations to confirm that correlation tests based on parsimony reconstructions of ancestral states behave themselves reasonably well statistically. D. Maddison (1994) concluded the effects of step-matrix bias were not all-obscuring. Slatkin and Maddison (1989) found that a parsimony-based statistic, the minimum number of migration events, was a reasonably good estimator of gene flow.

If one does estimate parameters of character evolution using historical reconstructions of change, then it is important to be concerned about these issues. It would be best to test for any possible biases. If you use MacClade's State Changes & Stasis chart, for instance, to estimate probabilities of various changes, you could try to set up null models and do simulations using MacClade's randomization and random data generation facilities to see how the parsimony reconstructions may be affecting your estimates (see the example given in the section "[Reconstructions used in State Changes & Stasis charts](#)" on page 397).

Multiple solutions

When one infers models using historical reconstructions, then ambiguity in the historical reconstruction can lead to ambiguity in the model inferred. On occasion, the data at hand will not speak for a single historical reconstruction. There may be equally well supported trees, or equally well supported reconstructions upon each tree (e.g., see D. Maddison, 1994).

Parsimony-based methods generally provide a score (cost or length) by which one can judge the relative merits of trees or character reconstructions. One can examine and select a collection of acceptable trees or character reconstructions based on the value of objective function. But the multiple solutions that often result create problems in analyzing character evolution, as different trees or character reconstructions may favor different notions about the history and models of character change.

Some believe it fruitful or necessary to find ways to eliminate some of the solutions (Brooks et al., 1986, Carpenter, 1988), but we believe it important to accept the diversity of solutions that are supported by the data. Unless one is to choose solutions arbitrarily, then multiple equally supported solutions will continue to be a common result of phylogenetic analyses. Indeed, we feel it important that the full diversity of equally supported solutions be consciously sought (D. Maddison, 1991b) especially as more is learned about confidence limits on phylogenies (Cavender, 1978, 1981; Templeton, 1983; Felsenstein, 1985b,c; Sanderson, 1989).

Tools need to be developed that can more readily examine multiple trees and multiple character recon-

structions, and summarize and present results to the biologist. One of MacClade's main goals from the start was to encourage biologists to accept that there is uncertainty in the phylogenetic tree and to explore character evolution under alternative trees, at first using the tree editor to change the tree by hand (W. Maddison, 1986; Maddison and Maddison, 1987) and later using charts that either resolve a polytomous tree or perform simulations of speciation (Maddison and Maddison, 1992). This encouragement was reiterated by D. Maddison (1994), Losos (1995), and Martins (1996). The importance of examining multiple solutions has been one of the prime motivating factors behind MacClade's charting facilities, which will examine multiple trees and character reconstructions, and behind MacClade's tracing features that indicate ambiguity. However, for aspects of character evolution beyond those examined by MacClade, few tools are currently available for automatic processing of multiple solutions. This is an important area for future development.

What assumptions are acceptable?

Let us return to the issues of assumptions and circularity. We have discussed hypotheses about processes of character change and how they could be both used as assumptions for phylogenetic analysis, and tested by phylogenetic information. There is a clear danger in such activities of assuming what one tries to show. In recognition of this danger, it has been urged that one should, as much as possible, avoid making assumptions about evolutionary processes while reconstructing evolutionary history, so that the reconstructed phylogenies can supply independent tests of evolutionary process theories (see Platnick, 1986 and references therein). Mitter (1981) notes that among zoological systematists "there is widespread (but not universal) agreement that ... systematic methods should be as free as possible from assumptions about how evolution works, because these assumptions are in general not testable without reference to systematic results." This sensible prescription might suggest that there should be an imbalance of trade between phylogenetic biology and evolutionary process theory — that phylogenetics should export trees to process theory for use there as assumptions, and yet be reluctant to import assumptions about the processes of evolution. (Reluctance to use process assumptions is by no means universal, however. See Felsenstein, 1988a; Harvey and Pagel, 1991.) Because MacClade gives you opportunities to use elaborate assumptions in opposition to this prescription, a word on the use of assumptions is in order.

Circularity can happen. One might assume a process, reconstruct the phylogeny, then use the phylogeny as evidence for the very process assumed. However, while process assumptions invite poor science, they do not require it. Care can be taken that the phylogeny is used to test process theories other than the ones that went into it. Thus, process theories used as assumptions place limitations on the uses of the reconstructed phylogeny (Platnick, 1986). The assumed process theories might concern different and better-established aspects of character evolution, or less specific aspects, than those derived from the study. In some circumstances, the process theories will merely provide a bias, like the step-matrix bias discussed earlier, that can to some extent be escaped. In the sequence of assumed process theory → tree → inferred process theory, the inferred process theory is no more forced to be the same as that assumed than are the trees at two stages of an iterative process like successive approximations character weighting (Farris, 1969) doomed to be identical.

What process theories can be legitimately applied? Lee (1999) argues that the danger of circularity or bias is greatest when the process theories are derived primarily from previous phylogenetic studies instead of from independent branches of biology. Maddison and Maddison (1992:68-71) suggest that the most important consideration is not so much circularity but whether the process assumptions are strong enough to be enforced when applied against one's data. Maddison and Maddison outline the conditions for process assumptions to have sufficient strength: they must be backed by enough evidence, and they must be sufficiently constraining or have sufficient ramifications that they cannot be easily dismissed as locally inapplicable by adding a few *ad hoc* assumptions. In such circumstances, parsimony not only allows use of the assumptions, but compells their use. The reader is referred to Hull (1967) for further discussion of circularity in systematic biology.

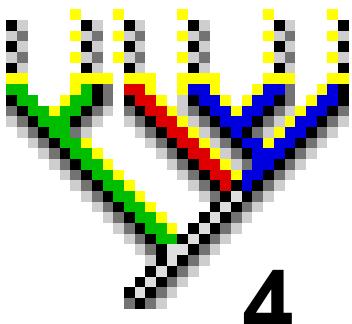
Parsimony versus other criteria

In MacClade, parsimony is used both in judging the phylogenetic tree and in reconstructing the history of character change. The justification of parsimony methods in phylogenetic biology has been a matter of much controversy (see summaries provided by Sober 1988; Steel and Penny, 2000; Swofford et al., 1996). However much we might hope that parsimony is merely a neutral methodological principle, it does guide us to choose some hypotheses over others, and it will guide us incorrectly in certain circumstances (Felsenstein, 1978b; Hendy and Penny, 1989). In general, we know that if evolution behaves according to some well-defined rules, parsimony methods will not be consistent estimators of phylogenies; we also know that there exist other models of evolutionary process under which parsimony methods will be reliable estimators (Felsenstein, 1979; Sober, 1988). All methods proposed to date would have their assumptions violated at least occasionally, and it remains to be seen how often they are violated and which method is most robust to violations in its assumptions.

Over the last decade there has been an increasing shift toward likelihood methods. Even if the consensus of the field is that likelihood methods are to be preferred in general, this does not mean that likelihood is necessarily the best choice in any particular case. For DNA data, for instance, statistical behavior may be well enough known and consistent enough across taxa and characters that fully statistical methods like maximum likelihood could be justifiably applied. In contrast, other characters may not fall into classes with known statistical behavior, and indeed their behavior may be the object of study. Many morphological characters whose genetic and developmental bases are not well understood might fall into this category.

We cannot simply ignore morphological, behavioral and other characters whose statistical behavior is not well understood. Despite their detractors, morphological characters persist in informing us about phylogenetic relationships (Hillis, 1987; Donoghue and Sanderson, 1992; W. Maddison, 1996). In addition, even if we were (1) to show convincingly that some molecular characters were well-behaved phylogenetically and (2) to decide always to place primary confidence in these molecular characters for phylogeny reconstruction, our goal is not merely a phylogenetic tree relating the few tens of thousands of specimens sequenced. When a specimen from a single species of a genus of orchids is chosen for sequencing, we will legitimately assume, on the basis of morphological characters, that other members of the species and the genus will fall with it on the phylogeny. Not only will we rely on morphology in this case, we need to rely on it: otherwise, our molecular phylogenies will be powerless to inform us about the evolution of the vast majority of specimens and species on earth that remain and will remain unsequenced. Molecular phylogenies can at best be thin skeletons on which the majority of organisms are arranged by morphological characters (W. Maddison, 1996). Finally, there is another reason we cannot ignore characters whose statistical behavior is not well known. Many of the most fascinating biological questions concern the evolution of morphological, behavioral, or other characters each of which has unique characteristics. We would be hard-pressed to justify assumption of an explicit statistical model of their evolution in order to reconstruct their history. For these characters, parsimony-based methods may always be a reasonable place to begin understanding their evolution. Of course, if robust nonparametric phylogenetic statistics can be developed, then perhaps, even for these characters, fully statistical methods may eventually be preferred.

Alternative methods such as parsimony and maximum likelihood need not always be kept separate. One could imagine likelihood methods applied to molecular characters to reconstruct the phylogeny, then parsimony used to interpret the evolution of morphological characters, in the light of the reconstructed phylogeny.



RECONSTRUCTING CHARACTER EVOLUTION USING PARSIMONY

4

The pathways and patterns shown by character evolution hold an important key to our understanding of evolutionary processes, as discussed in [Chapter 3](#). Could we see into the past, we would want to know which characteristics transformed to which others, how often, in what lineages and at what times, and in what contexts. Patterns in these events might indicate what forces cause characters to change or remain stable, including forces arising from other characters possessed. Of course, in most cases we will have to be satisfied with inferring the history of character evolution indirectly. With this aim, a number of methods have been developed to use reconstructed phylogenetic trees along with the observed character states of organisms to infer the states of ancestral lineages and their changes through time.

In this chapter we describe parsimony-based methods to reconstruct character evolution. After explaining exactly what we are trying to reconstruct, we will discuss separately the assumptions that specify how parsimonious a reconstruction is, and the algorithms that find what reconstructions are most parsimonious under those assumptions. We will describe the algorithms as they are implemented in MacClade, in order to document MacClade's calculations. The reader is referred to the previous chapter for additional discussion of parsimony reconstructions of character evolution.

Some statistics about trees and characters are derived from parsimonious reconstructions of character evolution. These include the number of steps in a character, the treelength, the consistency index, and so on. In this chapter only the calculations of number of steps and changes are described, because their calculations are intimately tied up with the reconstruction algorithms. The chapter "[Basic Tree and Character Statistics](#)" [on page 373](#) discusses treelength and other indices.

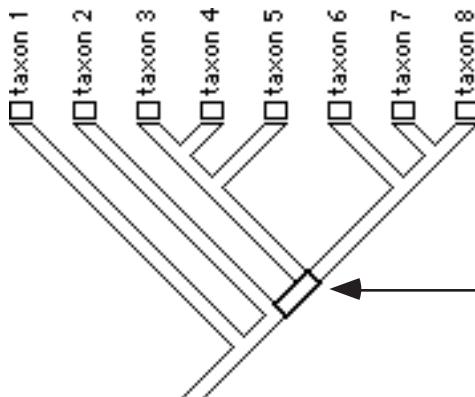
Goals, assumptions, and algorithms

In reconstructing character evolution, our **goal** is to reconstruct the character states of hypothetical ancestors throughout the phylogenetic tree ("[Inferring the history of character change](#)" [on page 54](#)). In particular, the usual practice is to estimate the states of the hypothetical ancestors at the nodes or branch points of the tree, because these provide convenient reference points (see [Chapter 3](#) for a discussion of nodes and branches). We will assume that character evolution is being reconstructed on a particular phylogenetic tree. That is, the phylogenetic tree has already been reconstructed, and is no longer an issue.

Under various **assumptions** about the nature of changes between states, including the number of steps between states, parsimony methods seek to reconstruct ancestral states by choosing the states at the branch points that require the minimal number of steps on the tree. Thus, parsimony together with the assumptions provides a **criterion** for choosing among character reconstructions.

But there are many possible reconstructions, and finding the reconstruction or reconstructions with the minimum number of steps is not always an easy task. Various **algorithms** (computational procedures) have been developed that accomplish this task of finding the most parsimonious reconstruction.

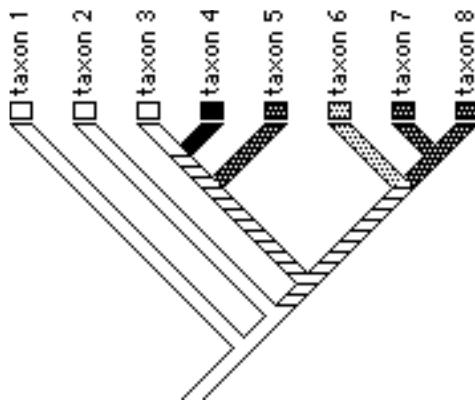
Ancestral states and reconstructions



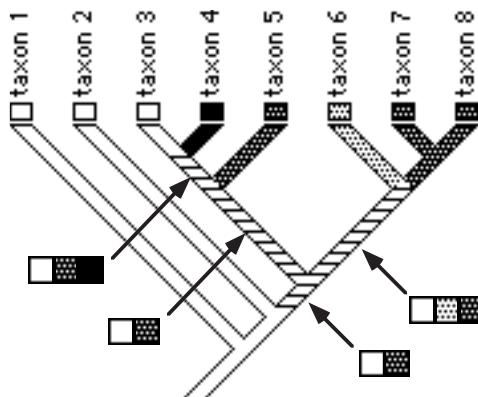
The result of our inference is an assignment of states to each node in the tree, indicating those states that can be placed at the node parsimoniously. If we had a phylogeny as shown above, with all taxa observed to have the white state, then obviously the assignment of the black state to the branch marked by the arrow is not parsimonious, because that would require one white-to-black change and two black-to-white changes. There is only one parsimonious assignment to this branch: the white state.

A state can be placed at the node parsimoniously if it allows the observed states in the taxa to be evolved in as few as possible evolutionary steps. If more than one state can be placed parsimoniously at a node, the node's assignment is said to be *equivocal*. The set of all the states that can be parsimoniously assigned to a node is called the node's **MPR set** (Swofford and Maddison, 1987; the name refers to the fact that these states occur in the most parsimonious reconstructions). The purpose of MacClade's Trace Character facility is to shade the branches to indicate the nodes' MPR sets.

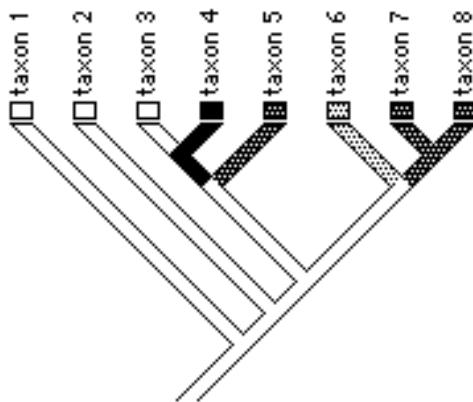
An equivocal assignment is indicated in MacClade by shading a branch with a horizontally striped pattern. For the unordered character shown traced onto the tree below, there are four branches shaded with the equivocal pattern (the pattern of horizontal lines):



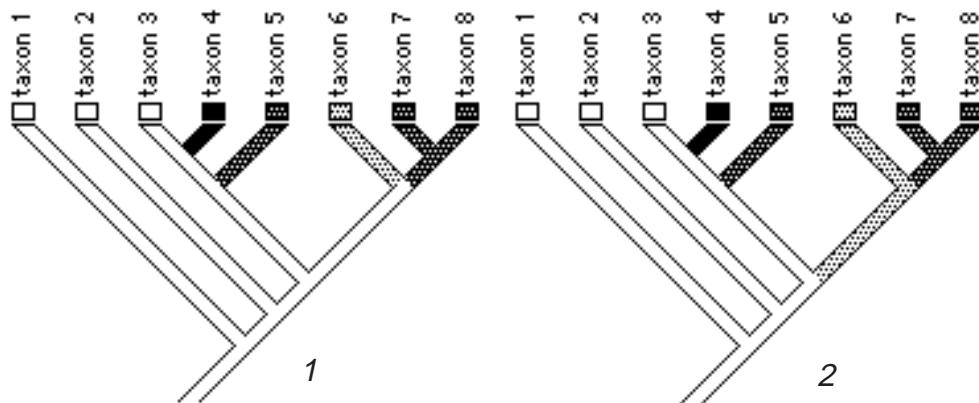
Two of the nodes have three states in their MPR sets, and two have two states:

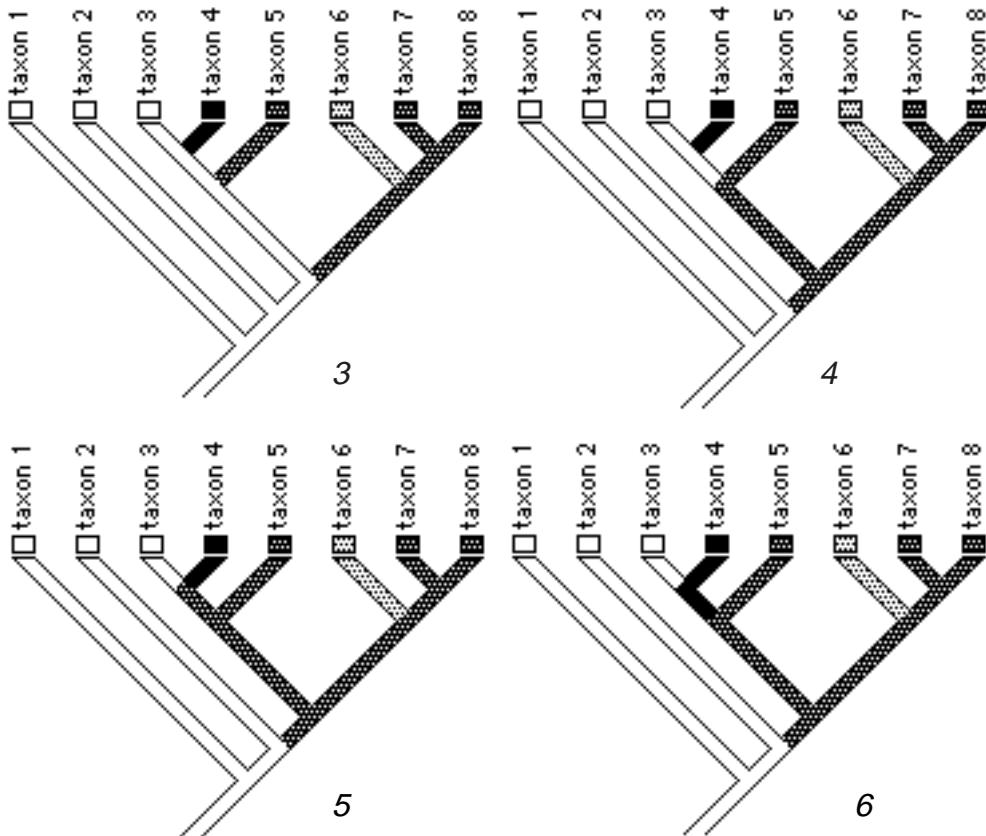


There happen to be six equally parsimonious **reconstructions** for this example; that is, there are six most-parsimonious resolutions of the equivocal regions in the tree above. (Other examples may have from one to billions of equally parsimonious reconstructions.) A reconstruction is a set of assignments to the nodes such that each node is assigned only one state. You can think of it as one particular scenario for the evolution of the trait in question. The diagram above doesn't directly tell you about such individual reconstructions, however. For example, looking at the above figure you might think that the following was a parsimonious reconstruction, because it chooses one state from the MPR set at each of the nodes:



In fact, this is not one of the parsimonious reconstructions. It indicates five evolutionary steps, but only four are needed. There are $36 (=2 \times 2 \times 3 \times 3)$ different reconstructions that can be obtained by choosing states from MPR sets, but only six of them are parsimonious. The six reconstructions that are parsimonious are:





Each one of these reconstructions shows four evolutionary steps. The problem with the first reconstruction we showed was that, although the one internal branch could be black, it could not parsimoniously be black at the same time that the other internal branches were white.

The meaning of "reconstruction" given above is a very specific one — a fully resolved (lacking ambiguity) scenario for the evolution of the character, node by node, on the tree.

The nature of a reconstruction of character evolution is discussed also on [page 54](#) and [Chapter 18](#).

Assumptions about character evolution

For a given character, various assumptions about transformation between states are possible. You may believe that the states follow a linear transformation series, or that gains are more difficult than losses, and so on. In order to select parsimonious reconstructions, we need to consider these assumptions, for they will influence how parsimony is quantified. It is not enough to say "minimize evolutionary steps", because perhaps transformations between some states involve more evolutionary steps than do others.

In MacClade these assumptions are specified by **transformation types**. A transformation type indicates certain rules about transformations between states. If you specify a character is of a particular type, MacClade will use that type's rules in reconstructing the character. For example, if you specify that a character is of the **ordered** type, then MacClade will assume the states evolve along a linear transformation series, whereas if you specify the character is of the **unordered** type, then MacClade will allow any state to transform to any other in a single step.

The transformation type assigned to a character will clearly affect both the reconstruction of character evolution, and the number of steps of evolution required for the character on the tree (see Swofford and Mad-

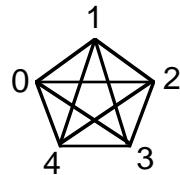
dison, 1992). For example, a reconstruction that requires fewer steps than any other under the assumption that a character is unordered may suggest changes of all sorts, from every state to every other state. Under the assumption that character states are ordered, some of these changes would require multiple steps, and this particular reconstruction might no longer be the most parsimonious.

These transformation types are essentially specifications of intra-character weightings, or the costs of presuming a particular type of change. Thus, if you assume a character to be ordered, you are weighting some state changes more highly than others. For example, a change in an ordered character between states 0 and 1 would count as one step, whereas a change between state 0 and 8 would be weighted eight times as much (that is, it would count as eight steps), and a change from 7 to 4 would count as three steps. If you assume a character to be unordered, you are weighting all state changes equally, such that a change between states 0 and 1, 0 and 8, or 7 and 4 each would count as one step. A brief introduction to choosing these weights is given in [Chapter 3](#).

Given the variety of weights that could be applied, many transformation types are possible. MacClade supplies five predefined transformation types: **unordered**, **ordered**, **irreversible**, **stratigraphic** and **Dollo**. These predefined transformation types can be characterized as follows.

Unordered

For characters designated as unordered, a change from any state to any other state is counted as one step ("Fitch parsimony"; Fitch [1971]; Hartigan [1973]). Thus, a change from 0 to 1, or from 0 to 8, or 7 to 4, are each counted as one step. A five-state unordered character can be represented diagrammatically as shown below, where change between any two states involves only one step (i.e., only one line has to be traversed in the diagram).



Nucleotide sequence characters might, as a first approximation, be considered as having unordered states, with a transformation from any one base A, C, G, or T to any other base being one step. Alternatively, step matrices may be used as described below to impose a higher cost for transversions.

Ordered

For characters designated as ordered, the number of steps from one state to another state is counted as the (absolute value of the) difference between their state numbers ("Wagner parsimony"; Farris [1970], Swoford and Maddison [1987]). Thus, a change from 0 to 1 is counted as one step, from 0 to 8 as eight steps, from 7 to 4 as three steps. A five-state ordered character can be represented diagrammatically as shown below.

0 — 1 — 2 — 3 — 4

In this diagram, the number of steps in the change between any two states is equal to the number of lines on the path between the two states; thus, from 1 to 4 is three lines or three steps. Morphological characters in which one state is intermediate between others (medium between small and large; 5 leaflets between 3 and 7) might reasonably be considered ordered characters.

Irreversible

For characters designated as irreversible, the number of steps from one state to another state is counted as the difference between their state numbers, with the restriction that decreases in state number do not occur

("Camin-Sokal parsimony"; Camin and Sokal [1965]). Thus, a change from 0 to 1 is counted as one step, from 0 to 8 as eight steps, but changes from 1 to 0 or 8 to 0 are impossible. Multiple gains are allowed, but no losses are allowed. A five-state irreversible character can be represented diagrammatically, as shown below.

$$0 \rightarrow 1 \rightarrow 2 \rightarrow 3 \rightarrow 4$$

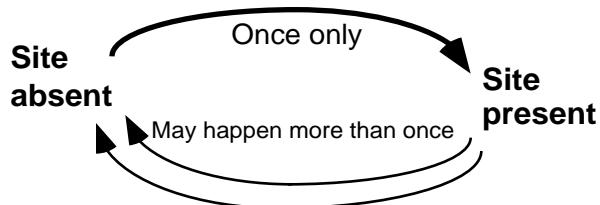
Stratigraphic

Characters assigned this type are treated as indicating stratigraphic age, with each state representing a stratum, based on Fisher's (1982, 1988, 1992) notion of stratigraphic parsimony. The oldest stratum is coded by state 0; younger strata are coded by successively higher states. The reconstruction of states at ancestral nodes that MacClade provides is therefore in some sense a reconstruction of the ages of ancestors. Clearly, descendants cannot be older than ancestors and so stratigraphic characters are irreversible. Ancestral ages are assigned using Fisher's (1992) criterion to minimize the instances in which the tree implies that a fossil should have been found in a stratum but wasn't. As a default, one step is counted for each absence in a stratum, but different costs can be assigned to absences in different strata. For instance, a lineage implied as missing from an extensive and well-known stratum might count more heavily against the tree than a lineage missing from a poorly known stratum. Further details on the interpretation of this character type are discussed below in the section on finding parsimonious reconstructions.

The stratigraphic character type allows paleontologists to determine how well a phylogenetic tree agrees or disagrees with stratigraphy, using a sort of "stratigraphic parsimony". A phylogenetic tree disagrees with stratigraphic data when it suggests fossils should occur in some strata but they have not yet been found there. Therefore, a parsimony-based approach might prefer those trees that minimize the amount of fossil record that is absent from the known stratigraphic record, but that the tree indicates should exist (i.e., that minimizes *ad hoc* hypotheses of missing fossils). Fisher's (1982, 1992) formal criterion is implemented in MacClade's stratigraphic character type. An early application of stratigraphic parsimony is that of Doyle et al. (1982).

Dollo

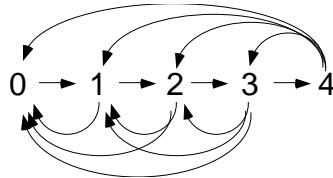
For Dollo characters, the number of steps from one state to another state is counted as the difference between the numbers given to the states, with the restriction that each increase in state number occurs only once (Farris, 1977). Thus, a change from 0 to 1 is counted as one step, from 0 to 6 as six steps, but gain of 1 or 6 can happen only once each on the tree. Multiple losses are allowed, multiple gains are not. One possible use for Dollo characters is in restriction site data (e.g., DeBry and Slade, 1985), where state 1 (site present) might be thought difficult to gain but easy to lose. Thus a two-state Dollo character for restriction site data might be represented as shown below.



MacClade 4 assumes, as did MacClade 3 (Maddison and Maddison, 1992), what Swofford (1991) calls the "unrooted" Dollo assumption. That is, MacClade does not assume that there was a gain from state 0 to state 1; it assumes only that there was no more than one gain from 0 to 1, no more than one from 1 to 2, and so on.

Step Matrices

In addition to the predefined transformation types discussed above, MacClade users can specify their own assumptions. You can do this by indicating the number of steps for a transformation from each state to each other state.



To:

| | 0 | 1 | 2 | 3 | 4 |
|-------|---|---|---|---|---|
| From: | 0 | 0 | 1 | 2 | 3 |
| | 1 | 1 | 0 | 1 | 2 |
| | 2 | 1 | 1 | 0 | 1 |
| | 3 | 1 | 1 | 1 | 0 |
| | 4 | 1 | 1 | 1 | 1 |

For instance, suppose you had an assumption that transformation between states followed the pathways indicated on the diagram above. This supposes that for increases in character state (which might be thought of as gains) the character behaves as an ordered character, with one step between 0 and 1, three steps between 0 and 3, and so on. For any decrease in character state, there is one step. Thus, you may believe gains have to be accumulated, but any loss can be accomplished in a single step (perhaps by deletions or modifiers). You can take this diagram and translate it into the matrix above. This matrix claims that there are no steps between 0 and itself, four steps from state 0 to state 4, one step from 4 to 0, and so on.

This matrix format for indicating distances between states and the algorithm that uses it (Sankoff and Rousseau, 1975; Sankoff and Cedergren, 1983; Maddison and Maddison, 1987) are very flexible, and with it you can code all sorts of elaborate assumptions such as asymmetry in number of steps between gains and losses, and so on. The predefined transformation types unordered, ordered and irreversible can be written in this format:

| unordered | | | | | | ordered | | | | | | irreversible | | | | | |
|-----------|---|---|---|---|---|---------|---|---|---|---|---|--------------|---|---|---|---|---|
| 0 | 1 | 2 | 3 | 4 | 5 | 0 | 1 | 2 | 3 | 4 | 5 | 0 | 1 | 2 | 3 | 4 | 5 |
| 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 2 | 3 | 4 | 0 | 0 | 1 | 2 | 3 | 4 |
| 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 2 | 3 | 4 | ∞ | 0 | 1 | 2 | 3 |
| 2 | 1 | 1 | 0 | 1 | 1 | 1 | 2 | 2 | 1 | 0 | 1 | 2 | ∞ | ∞ | 0 | 1 | 2 |
| 3 | 1 | 1 | 1 | 0 | 1 | 1 | 3 | 3 | 2 | 1 | 0 | 1 | 2 | ∞ | ∞ | ∞ | 0 |
| 4 | 1 | 1 | 1 | 1 | 0 | 1 | 4 | 4 | 3 | 2 | 1 | 0 | 1 | ∞ | ∞ | ∞ | 0 |
| 5 | 1 | 1 | 1 | 1 | 1 | 0 | 5 | 5 | 4 | 3 | 2 | 1 | 0 | ∞ | ∞ | ∞ | ∞ |

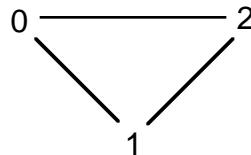
In [Chapter 3](#) is a discussion of the source of such step matrices.

Violating the triangle inequality

It is possible to define a matrix of state-to-state distances that disobeys a fundamental property of dis-

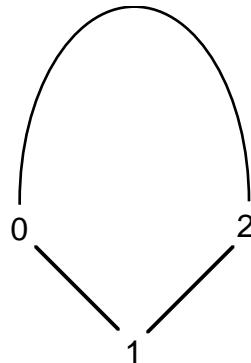
tances called the "triangle inequality". The name is derived from the fact that triangles in Euclidean space have the property that the length of one side is always less than the sum of the lengths of the other two sides. More generally, the triangle inequality states (more or less) that the shortest distance between two points is a straight line. For instance suppose the direct distance from state i to j is more than the distance from i to k plus k to j . This means that the shortest distance between two states is not a straight line; that is, there is an indirect route that provides a shortcut. A transformation type matrix that contains such a short-cut is said to violate the triangle inequality.

For example, in the illustration below, distances between states satisfy the triangle inequality, as the distance from 0 to 1 and then to 2 is not shorter than the distance from 0 directly to 2.



Distances that satisfy
the triangle inequality

However, in the illustration below, the triangle inequality is violated, as the distance from 0 to 1 and then to 2 is shorter than the direct route from 0 to 2.



Distances that violate
the triangle inequality

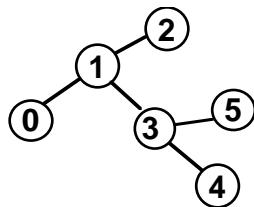
Whether or not a matrix implying such distances is nonsensical and internally inconsistent depends on exactly how you interpret it. Suppose the matrix says the distance from state i to j is 5. If you take this to mean that a transition from i to j costs five steps regardless of the time taken to accomplish the transition, 10 years or 10,000,000 years, then the matrix must be considered self-inconsistent. It is claiming that one can go between i and j in a certain number of steps whereas in fact the matrix itself allows a more parsimonious transition through an intermediate. However, if the matrix's claimed cost for a transition is qualified by a time limitation, the inconsistency can be avoided. For example, a change between two states i and k may be unlikely over a short time period (perhaps the length of a branch on the tree), in fact much less likely than an i to j or j to k change. But over a longer time interval, the likelihood of an i to j change followed by a j to k change makes an i to k change more likely. Thus, if the matrix was presumed to reflect the probability of changing over the time span of short branches, violation of the triangle inequality would not make the matrix self-inconsistent. This latter view of a step matrix is rather different from the traditional parsimony view in which time is no factor. Whenever there are violations of the triangle inequality, however, it would seem we are forced into this view if we want to avoid inconsistency. If we want to maintain

a standard parsimony view in which time is not considered, then we must accept the inconsistency of step matrices that violate the triangle inequality.

Violations of the triangle inequality are particularly troublesome when terminal taxa are polymorphic. The algorithms for reconstructing ancestral states count steps from one end of a branch to the other; they do not imagine interpolated changes along a branch, with the exception of changes within a polymorphic terminal taxon. When a terminal taxon is polymorphic, MacClade allows changes to occur *within* the terminal taxon, even though there is no "branch" there. If you view a step matrix as indicating the cost of changes over the time span of a branch, which you might do to avoid inconsistency when the triangle inequality is violated, it is not clear how costs are to be assessed to implicit changes within terminal taxa.

Character state trees

Assumptions about the evolution of a character can also take the form of a character state tree. The tree below, for instance, says that to evolve from state 1 to state 5, state 3 is a necessary intermediate. In MacClade, character state trees are reversible and unrooted, meaning that transformations from 5 to 3 are also possible.



A character state tree

To some people a character state tree such as that shown above means that character state 0 was ancestral, and it then evolved to state 1, then 1 evolved to 2 and 3, and 3 evolved to states 5 and 4. Such an interpretation would be that the tree directly shows the actual history of character change. An alternative interpretation is that it shows only the possible pathways that evolution could have followed. Thus the tree says that if evolution were to proceed from state 0 to state 5, it would have to pass through states 1 and 3. Evolution could go from state 4 to state 5, but would have to pass through state 3. State 0 need not even be ancestral.

Of the two interpretations, clearly the possible-pathways model (see [Chapter 3](#)) is more appropriate for our purposes than the history of change. Were we to suppose we already knew the history of character change, there would be little point in reconstructing character evolution using the character state tree as an assumption. Instead, the character state tree as an indication of possible pathways is perfectly appropriate as an assumption to constrain (but not completely specify) our conclusions about character evolution.

Ancestral polymorphism

MacClade's algorithms used for reconstructing character evolution assume that for each character there is only one state at each point on the tree's branches. That is, ancestral species are presumed to be monomorphic. The only place where polymorphism is allowed is in the terminal taxa (although see [Chapter 3](#) and the cautions of Nixon and Davis, 1991). For ancestral branches MacClade sometimes shows an equivocal pattern representing more than one state, but this does not mean that such a branch is polymorphic. Rather, it means that any of the states represented could by itself be placed on that branch with equal parsimony.

The implicit assumption that ancestors can have only one state at a time may or may not be appropriate. It is perhaps most questionable for molecular characters in which a polymorphic condition is quite conceivable and may even be observed. For instance, suppose that two sister species are observed both to be polymorphic at a nucleotide site, both having A and G. The unordered character parsimony algorithm will

presume that their immediate common ancestor was monomorphic, either for A or G, and that the alternative base was independently derived in the two species. In such a situation we might prefer to suppose the ancestor was also polymorphic and the polymorphism was simply maintained in the two daughter species.

The assumption of ancestral monomorphism is rather harsh, but it does simplify parsimony calculations. If ancestral polymorphisms were allowed, one could explain any character distribution by supposing a highly polymorphic ancestor with successive loss of states, or by supposing convergence, or in many other ways. Thus one would have to balance gains, retention of polymorphism, and loss in deciding which reconstruction of evolution is preferable.

Polymorphic ancestors have been incorporated into parsimony reconstructions of ancestral states by Farris (1978) and Felsenstein (1979), who proposed parsimony calculations that minimize fixations after a single gain of a polymorphic condition, and by Cavalli-Sforza and Edwards (1967), Rogers (1984), and Swofford and Berlocher (1987), who minimized amount of gene frequency change. The latter criterion uses data (gene frequencies) very different from those used in MacClade. Neither the "polymorphism parsimony" of Felsenstein and Farris or the "frequency parsimony" of Swofford and Berlocher is available in MacClade 4.

Maddison and Maddison (1987; included in "Read Me First" file on disk in September 1987, and in manual after January 1989) proposed a method to allow ancestral polymorphisms using step matrices. They suggested recoding the data to represent *population or species states* instead of organismal states (see "["Polymorphisms and missing data" on page 43](#)"). For instance a binary character that could exist monomorphically as states 0 or 1, or polymorphically as [0,1] gets recoded so that its new states are 0 and 2 for the monomorphic cases and 1 for the polymorphic case. Then one can choose or make a transformation type that indicates how many steps are required between one population state and any other. Perhaps the simplest assumption is that there is one step for any gain or loss of a state and no penalty for retaining polymorphisms; the recoded character could then be treated as ordered. This logic can be extended to a three- and four-state character. (For more than four organism states, the number of population states becomes larger than 26, the most states allowed by MacClade.) If one believed it to be one step for any gain of a state and one step for any loss, with no penalty for retaining polymorphisms, a user-defined type with the following matrix of steps could be used for a three-state character:

| To: | 0 | 1 | 2 | 01 | 02 | 12 | 012 |
|-------|-----|---------------|---|----|----|----|-----|
| From: | 0 | 0 2 2 1 1 3 2 | | | | | |
| | 1 | 2 0 2 1 3 1 2 | | | | | |
| | 2 | 2 2 0 3 1 1 2 | | | | | |
| | 01 | 1 1 3 0 2 2 1 | | | | | |
| | 02 | 1 3 1 2 0 2 1 | | | | | |
| | 12 | 3 1 1 2 2 0 1 | | | | | |
| | 012 | 2 2 2 1 1 1 0 | | | | | |

Thus it is 2 steps to go from condition 0 (monomorphic for 0) to 012 (polymorphic for 0, 1, and 2), but 3 steps to go from 0 to 12 because states 1 and 2 would need to be gained and state 0 lost. To implement this in MacClade, a new recoded population character could be made that codes the population states 0, 1, 2, 01, 02, 12, and 012 to be represented by states with symbols A, B, C, D, E, F, and G (for instance).

The original organismal character might lead one to superimpose various weights to changes. For instance, the step matrix for population transformations might be:

| To: | 0 | 1 | 2 | 01 | 02 | 12 | 012 |
|-------|-----|---------------|---|----|----|----|-----|
| From: | 0 | 0 2 4 1 3 3 2 | | | | | |
| | 1 | 2 0 2 1 3 1 2 | | | | | |
| | 2 | 4 2 0 3 3 1 2 | | | | | |
| | 01 | 1 1 3 0 2 2 1 | | | | | |
| | 02 | 3 3 3 2 0 2 1 | | | | | |
| | 12 | 3 1 1 2 2 0 1 | | | | | |
| | 012 | 2 2 2 1 1 1 0 | | | | | |

if the original three-state organismal character was constrained to evolve in an ordered sequence. That is, if to change from state 0 to state 2, one had to go via state 1, then to go from population state 0 to state 02, three steps are required: first, evolve 1 from 0; then, evolve 2 from 1; then, lose 1. This would be appropriate, for instance to model the transition from an XXO species to an XXXY species by two chromosome fusions (W. Maddison, 1982): first, one fusion occurs to give XXY, then another to give XXXY, and the intermediate XXY and the original XXO need to be lost.

Users interested in exploring other ideas about retained polymorphism may want to investigate some of the options of Felsenstein's (1993) PHYLIP computer package or Swofford and Berlocher's (1987) FreqPars.

Finding the most parsimonious ancestral states

The problem of reconstructing the character states of ancestors using the principle of parsimony can be phrased as the following question: Given the form of the phylogenetic tree, the states observed in the terminal taxa, and the assumptions regarding character evolution, what assignments of states to the internal nodes of the tree require the fewest evolutionary steps? In this section we will describe algorithms to find what states can be parsimoniously assigned to each node of a tree. We will not consider algorithms to resolve individual reconstructions in cases of ambiguity; these are described in the following section.

In some cases it may be intuitively obvious how states can be most parsimoniously assigned to ancestors. For instance if all the extant members of one clade share state 1, then it is simplest to suppose all of the ancestral members of the clade also had state 1. In other cases it may not be so easy. Thus a number of algorithms have been developed to find the most parsimonious reconstruction of ancestral states on a tree (Farris, 1970, 1977, 1978; Fitch, 1971; Fitch and Farris, 1974; Hartigan, 1973; Moore et al., 1973; Sankoff and Rousseau, 1975; Wagner, 1981; Rogers, 1984; Sankoff and Cedergren, 1983; Swofford and Berlocher, 1987; Swofford and Maddison, 1987; W. Maddison, 1989, 1991; Rinsma et al., 1990). Some of these are reviewed by Swofford and Maddison (1992). These different algorithms have been written to handle different restrictive assumptions imposed on character evolution. For instance, the algorithm of Farris (1970), completed by Swofford and Maddison (1987), reconstructs character evolution under the assumption that states are linearly ordered.

In the following sections the algorithms, as formulated in MacClade, are described. In these descriptions we will imagine that we are assigning states to the nodes of the tree. The nodes are the branching points; between them are the internodes or edges (see [Chapter 3](#)). Graphically, MacClade colors a node and the internode beneath it together as a single unit, though a change between one node and the next could have occurred anywhere along the internode between them.

The algorithms of MacClade are exact in that they find (barring bugs) all and only the most parsimonious assignments of character states that can be made to every node of the tree. There are two known exceptions to this: for user-defined type characters in which some taxa are polymorphic with more than two states, and for characters in which polymorphic observed taxa are made ancestral, MacClade may not give all most-parsimonious assignments. These problems are discussed below.

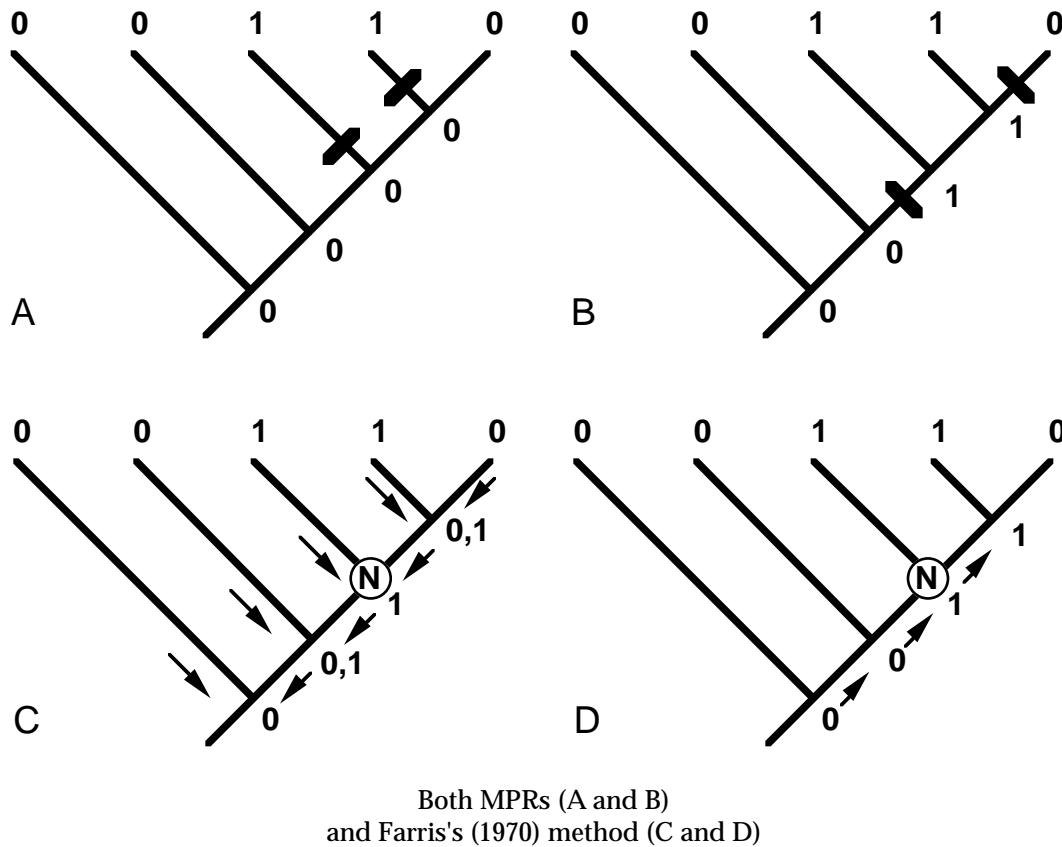
What information is used?

A parsimony reconstruction of character evolution uses three sources of information: the phylogenetic tree, the character states observed in the extant and fossil taxa, and the assumptions about character evolution. Whether the whole phylogenetic tree, or only part of it, is used in reconstructing ancestral states has been a matter of some confusion.

In phylogenetic systematics a distinction is made between the ingroup or study group, consisting of the descendants of a particular ancestor, and the outgroups, consisting of groups closely related to but not descended from this ancestor (e.g., Watrous and Wheeler, 1981). In order to determine which traits are derived (apomorphic) and thus indicate monophyletic groups within the ingroup, the states of outgroups have been classically used to reconstruct ancestral states (Stevens, 1980; Watrous and Wheeler, 1981; Farris, 1982; Maddison et al., 1984). Outgroup analysis has been a popular method to determine character polarity for this purpose, and so it is not surprising that its use has been extended to trace pathways and patterns of character evolution in general (e.g., Ridley, 1983; Mooi, 1989). These authors have suggested that one could reconstruct ancestral states by applying outgroup analysis stepwise, working from the bottom of the tree toward nodes higher and higher up. In our context, one could think of outgroup analysis as basing the state reconstruction of an internal node entirely on information from the node's sister group and cousins, that is, from terminal taxa below and beside the internal node, without any information from the node's descendants. In terms of the algorithms, using sequential outgroup analysis to estimate ancestral states would be like using only the states from the "uppass" of the algorithms (see "How the algorithms work", below). Although such an estimate may be adequate for special purposes, such as to help obtain a parsimonious reconstruction of phylogeny within the ingroup (Maddison et al., 1984), it is not adequate in general as an estimate of the ancestral state because it fails to take into account evidence from the descendants of the internal node of interest (Maddison et al., 1984:89).

One might take an exactly opposite approach — reconstruct a node's state using only information from its descendants. After all, if one tests a reconstruction by comparing its implications about what data are expected against observed data, then one might argue that the state at an internal node has implications only about its descendants, because only these are biologically derived from the ancestor. However, this would be confusing biological derivation with logical derivation. An outgroup's (cousin's) states give us evidence about the state at an internal node, because the states at the internal node's sisters and cousins might suggest a state at the internal node's ancestor, which might in turn suggest states at the internal node itself.

The states of both descendants and cousins can influence what is the parsimonious reconstruction of an internal node's states. The need to consider information evenly from above and below a node is well illustrated by the behavior of Farris's (1970) algorithm, which finds only some of the most parsimonious reconstructions for ordered characters (Swofford and Maddison, 1987). In the example below, there are two equally parsimonious reconstructions, one in which state 1 is gained twice independently (tree A) and another in which state 1 is gained then later lost (tree B).

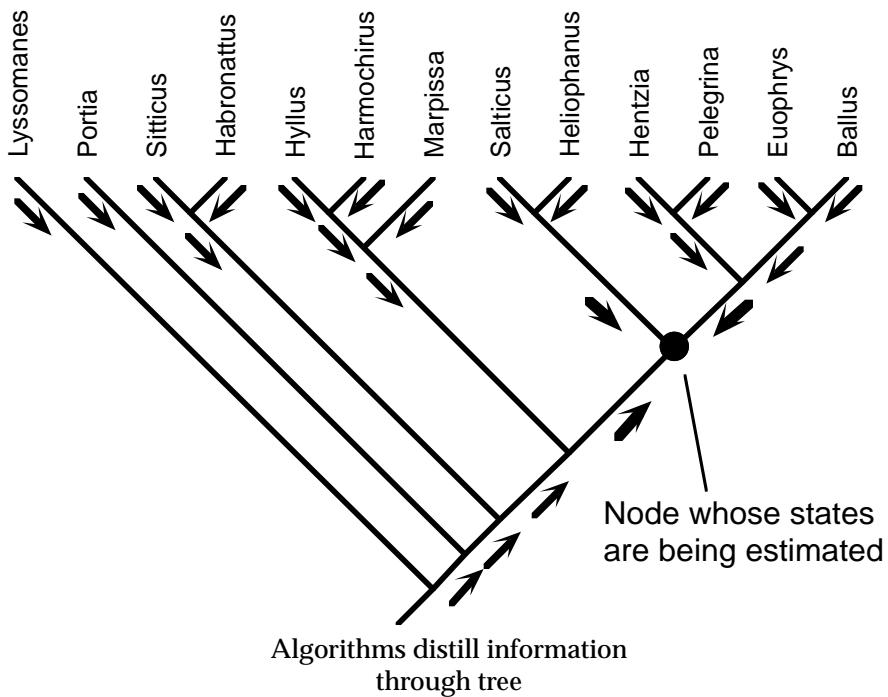


Farris's (1970) algorithm first proceeds downward on the tree, assigning sets of states to the nodes according to the preferences of the clade above the node, as shown in tree C. In particular, note that state 1 is assigned to the node N. Because the pass back up the tree (tree D) can only resolve ambiguities remaining from the downpass, information from below node N has only a secondary role in determining N's state because it cannot overrule the decisive assignment of 1 to N. The priority given to the clade above N prevents the discovery of equally parsimonious compromises that are disfavored above N but favored by the part of the tree below N. Information from all parts of the tree needs to be considered evenly for all optimal reconstructions to be found, at least under many assumptions of character change.

Because MacClade considers the whole tree when reconstructing ancestral states at any given node, it does not shade the outgroup node the same as would be achieved using the "outgroup algorithm" of Maddison et al. (1984), which considers the outgroup in isolation from the ingroup (see Maddison et al., 1984:88).

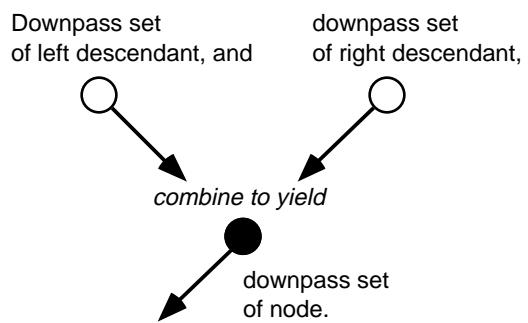
How the algorithms work

Each of these algorithms works in a similar way. Each moves through the tree, distilling information as it goes, in order to arrive at estimates for the states at each node. Information from above and below the node is distilled and combined to yield the estimate. Thus the state estimated for a node will depend on the information distilled (1) down through the clade of the node's left descendant, (2) down through the right descendant's clade, and (3) up from the part of the tree below and beside the node. These three sources of information converge at the node and indicate its state, as shown below.



In practice this distilling process is done in two or three passes up and down the tree. The details of the process differ with the different assumptions used. As an example, let us choose the algorithm for unordered characters.

The algorithm first makes a pass down the tree from the terminal taxa to the root (we will call it the **down-pass**). As it proceeds down the tree, the algorithm visits each node and calculates a set of states that can be appropriately assigned to that node. This set will be called the **downpass set** of the node, because this is the set of states preferred by that part of the tree above the node. To calculate this set of states, the algorithm uses the sets already calculated from the two nodes immediately above (descendant from) the node being visited.



Combining sets in downpass

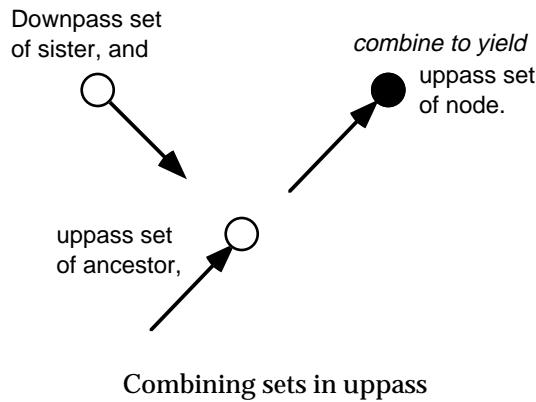
The unordered algorithm uses the following calculation: A node is assigned the set of states that are present in both state sets of the two descendant nodes if there are any such; if no states are present in both sets above, the node is assigned the union of the sets of the two descendant nodes. Once the state set has been calculated for the node being visited, the algorithm can then move down the tree to the node's immediate ancestor and perform the calculation on it. The algorithm continues down until it eventually arrives at the root. The downpass set at a node can be seen by holding the Option key and touching the list-states

tool on the branch when a character is traced in MacClade.

The states assigned to a node on the first downpass of the algorithm are not necessarily the most parsimonious assignments to the node according to the whole tree. When we arrive at a node on the downpass, only the nodes descendant from it have been used in the calculation of the visited node's states. Thus, the states assigned on the downpass are most parsimonious according to the clade above it, but may not be most parsimonious according to the whole tree because the state calculations have not yet considered nodes below the visited node. There is one exception to this result, in that there is one node for which the downpass states are most parsimonious. By the time the algorithm gets to the root, it has considered the entire tree, and the states assigned to the root are indeed the most parsimonious states according to the whole tree.

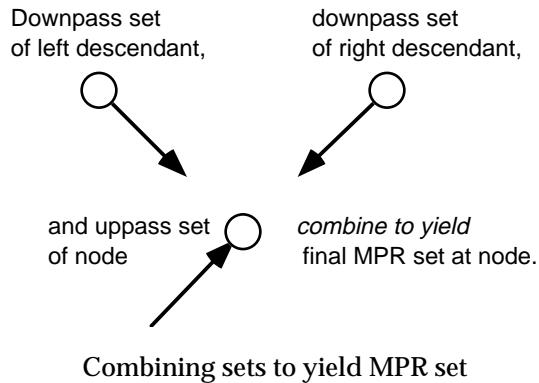
We want to know eventually, for every node, the set of states that can be most parsimoniously assigned to it according to the whole tree. Because the downpass algorithm guarantees whole-tree parsimony for the states assigned to the root, one way to achieve our goal would be to reroot the tree at each node, each time performing the downpass algorithm ending up at the new root, and thus finding the whole-tree parsimony state sets for each node. Effectively, MacClade does this, but there is a much more efficient way besides actually rerooting the tree and performing a whole new downpass each time (Swofford and Maddison, 1987; Swofford and Maddison, 1992).

After doing a single downpass, MacClade does another pass going up the tree (the **uppass**). The set of states assigned on the uppass will be called the **uppass set** of the node. This is the set of states preferred by that part of the tree below the node. The uppass state set of a node is calculated from the uppass set of the node below it (its ancestor) with the downpass set of the node beside it (its sister node), using the same calculation for combining state sets as for the downpass.



The root has no uppass set. A node immediately descendant from the root gets its uppass set directly from the downpass set of its sister, because its ancestor (the root) has no uppass set. The uppass set at a node can be seen by holding the Option key and touching the list-states tool on the branch when a character is traced in MacClade. If {1} is indicated as the uppass set, then the node is most parsimoniously assigned state 1 according to the part of the tree below this node that consists of the node's immediate ancestor, the node's sister node, and parts further away.

Once this second pass has been completed, we now have information distilled from all parts of the tree surrounding any node. For each node we know the opinion of its right-descendant clade as to what states the node should take (the downpass set of the right-descendent node), we know the opinion of its left-descendant clade (the downpass set of the left-descendent node), and we know the opinion from below the node (the uppass set of the node itself). These three opinions are then combined in a calculation that results in the final state set for the node (Swofford and Maddison, 1987). This calculation condenses three state sets into one.



For unordered characters, it chooses those states occurring in the greatest number of the three sets. The set of states assigned is the set of all most parsimonious states for the node, and will be called the **MPR set** of the node. We will emphasize the fact that it is found by the final pass of MacClade's algorithms by occasionally calling it the final MPR set of the node.

As MacClade is doing its calculations, it displays different cursors to indicate at what stage of the calculations it is in. When it is finding which states occur in the taxa in the tree, it shows . is shown when assigning downpass sets to terminal nodes. For some character types, cursors , , and are displayed for the downpass, uppass and final passes, respectively. For step-matrix type characters (including those with character state trees), MacClade displays , , and for the downpass, uppass, and final passes, respectively. While calculating the minimum or maximum conceivable number of steps in the characters, it shows .

Polytomies

When a polytomy is encountered, the algorithm needs to combine information from multiple nodes to yield the information placed at the polytymous node. The combining methods are described below in the context of each character transformation type. The method differs depending on whether the polytomy is treated as multiple speciation ("hard") or uncertainty ("soft"), as discussed by W. Maddison (1989) and below. The reader should refer to the section on interpreting character tracing (["States of nonterminal branches" on page 347](#)) and the discussion of polytomies (["Polytomies and their difficulties" on page 105](#)) for comments on the interpretations of polytomies.

The calculations on polytomies are particularly slow, because MacClade moves through the polytymous region to survey the multiple nodes separately for each character.

Designating observed taxa as ancestors

When observed taxa are designated as ancestors, MacClade assumes that the states observed in the taxon are directly on the internal node, except for stratigraphic characters, where special rules apply. For unordered, ordered, and irreversible characters, if the taxon is monomorphic or polymorphic, its states are forced to lie directly on the internal node N. (As noted in [Chapter 17](#), an observed taxon fixed as an ancestor can have only one descendant in MacClade.)

If the observed ancestral taxon had missing data or uncertain states, then N's states are flexible, being assigned in part according to the preferences of surrounding nodes. On the downpass, N acts as a filter of the downpass set from its descendent node. If N's taxon had missing data, the downpass set of N's descendent node is passed directly into N's downpass set. If N's taxon was of uncertain state, then N is assigned a downpass set consisting of those among the possible states at N that are closest to the downpass set of N's

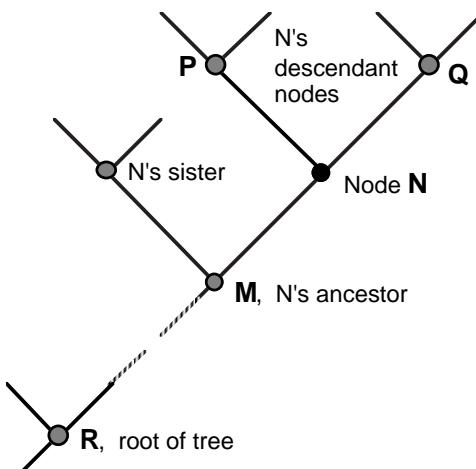
descendant. On the uppass, N acts similarly as a filter if its states were uncertain or unknown. On the final pass, the observed taxon, if of uncertain state, is treated temporarily as if it were not set in ancestral position, and N's states assessed using standard algorithms for combining preferences from the observed taxon, its sister node, and N's ancestor. This preliminary state set indicates the preferences from around the tree. N's set is then assigned to include those states in the taxon's possible observed states that are closest to the preliminary state set.

Counting steps

The descriptions of the algorithms focus on the problem of finding the MPR sets. However, there are some notes in each algorithm description regarding how the minimum number of steps in the character is counted. The problem of counting steps is discussed toward the end of this chapter, but notes are included here because it is much easier to describe the counting process in the context of the reconstruction algorithms.

Terms

In the following few sections we will give details about the exact calculations used in the algorithms. Generally, the node being discussed will be called N, its two descendent nodes P and Q (assuming N is dichotomous), and its ancestor M. The root of the whole tree will be called R.



The downpass set at N will be called D_N , the uppass set U_N , and the final MPR set F_N . A set of states including only the state t will be indicated as $\{t\}$; a set of states consisting of states a through b inclusive will be indicated as $\{a \dots b\}$. The number of elements in a set T is $\text{card}(T)$. The smallest element in T will be called $\text{min}(T)$; the largest element $\text{max}(T)$. The set $\{\text{min}(T) \dots \text{max}(T)\}$, containing all elements from the smallest in T to the largest, will be called $\text{range}(T)$. The set $\mathbf{D}(A,B)$ will consist of AB if it is not empty; otherwise, it is the element in A closest to B .

Unordered characters

Algorithms for unordered characters were first developed and proved by Fitch (1971) and Hartigan (1973); the algorithm is thus occasionally called "Fitch optimization". Hard polytomy algorithms are due to Hartigan; soft to W. Maddison (1989).

As mentioned above, the algorithms combine state sets of nearby nodes. For unordered characters, designate the combining function by \oplus . The combination of two sets A and B, designated $\oplus(A,B)$, is the intersection of A and B if it is not empty; otherwise, it is the union of A and B. The combination of three sets A, B, and C, designated $\oplus(A,B,C)$, is the intersection of A, B, and C, if it is not empty; otherwise, it is the union of pairwise intersections of the three sets $[(A \cap B) \cup (A \cap C) \cup (B \cap C)]$, if this is not empty; otherwise, it is the

union of A, B, and C.

Unordered

Assignment of downpass set to terminal nodes (card (T) is the number of states in T):

| State observed for terminal taxon | Downpass set assigned to terminal node | Steps counted |
|--|---|---------------|
| One observed state, t | {t} | 0 |
| Observed set of states T, uncertain | T | 0 |
| Observed set of states T, polymorphic | T | card(T)-1 |
| Missing data | {0...maxstate} | 0 |

"Maxstate" means the maximum observed or allowed state in the character. (This is not the same as the MAXSTATE option referred to elsewhere in the manual.)

Assignment of downpass set to internal node N based on sets at descendent nodes:

When N has two descendants P and Q, with downpass sets D_P and D_Q : Assign $\oplus(D_P, D_Q)$. Steps: One step is counted if a union was required in \oplus .

When N is polytomous, with k descendants with downpass sets $D_1 \dots D_k$:

"Soft" polytomy: Assign union of the smallest sets covering $D_1 \dots D_k$, where a state set S is said to cover $D_1 \dots D_k$ if S has a nonempty intersection with each of $D_1 \dots D_k$. Steps: m-1 steps are counted where m is the number of states in a smallest covering set.

"Hard" polytomy: Assign set consisting of those states occurring most frequently among $D_1 \dots D_k$. Steps: k-m steps are counted where m is the number of descendent nodes with one of the most frequent states.

When N is an observed taxon fixed as an ancestor, with descendant with downpass set D_P :

When N's observed set of states T is polymorphic or monomorphic: Assign T.

When N's observed set of states T is marked as uncertain: Assign $T \cap D_P$ if it is not empty; otherwise, assign T.

When N has missing data: Assign D_P .

Steps: Except if N has missing data, one step is counted whenever TDP is empty.

Assignment of uppass set to N:

Use same calculations as for downpass, except that when N's ancestor is dichotomous, combine N's ancestor's uppass set with N's sister's downpass set to yield N's uppass set; when N's ancestor is polytomous, combine N's ancestor's uppass set with downpass sets of all of N's sisters to yield N's uppass set; when N's ancestor M is an observed taxon fixed as an ancestor, N's uppass set is calculated using M's observed states and U_M just as T and D_P were used, respectively, in the downpass with taxon fixed as ancestor. (Because the root does not have an uppass set, if N's ancestor is the root, then assign for N's uppass set simply its sister node's downpass set.)

Assignment of final MPR set to terminal node N with observed set T and ancestor M:

When M is a dichotomous or hard polytomous node:

When T is monomorphic: Assign T.

When T is polymorphic: Assign F_M if F_M is a subset of T; otherwise, assign $T \cup F_M$.

When T has uncertain states: Assign F_M if F_M is a subset of T; otherwise, assign T.

When N had missing data: Assign F_M .

When N's ancestor is a soft polytomous node:

When T is monomorphic: Assign T.

When T is polymorphic, or with uncertain states: Assign $\oplus(T, U_N)$.

When N had missing data: Assign U_N .

Assignment of final MPR set to internal node N based on sets at surrounding nodes:

When N is the root: Assign N's downpass set to be the final MPR set.

When N has uppass set U_N and two descendants P and Q with downpass sets D_P and D_Q : Assign $\oplus(D_P, D_Q, U_N)$.

When N has uppass set U_N , downpass set D_N and is polytomous, with k descendants with downpass sets $D_1 \dots D_k$:

"Soft" polytomy: Assign $\oplus(D_N, U_N)$. (Recall node is below polytomy; see [page 348](#).)

"Hard" polytomy: Use the same calculations as for hard polytomies in downpass, except use sets $D_1 \dots D_k$ and U_N .

When N is an observed taxon fixed as an ancestor, with uppass set U_N , and with descendant P with downpass set D_P :

When N was observed monomorphic or polymorphic: Assign observed state(s).

When N had uncertain states, set T: Assign $T \cap \oplus(T, D_P, U_N)$ if it is not empty; otherwise, assign T.

When N had missing data: Assign $\oplus(D_P, U_N)$.

Ordered characters

Farris's (1970) algorithm ("Farris optimization" or "Wagner parsimony") for ordered characters found only some of the most parsimonious assignments (see [page 78](#)); it was completed and proved by Swofford and Maddison (1987). The ordered algorithm in MacClade is basically that of Swofford and Maddison. Polytomy algorithms are due to W. Maddison (1989).

As mentioned above, the algorithms combine state sets of nearby nodes. For ordered characters, designate the combining operator by \otimes . It is assumed that each state set contains all states between any two of its states; that is, there are no gaps in the set. (If there are, convert the set to its own range.) The combination of two sets A and B, designated $\otimes(A, B)$, is the intersection of A and B, if it is not empty; otherwise, it is $\{\max(A) \dots \min(B)\}$ if $\min(B) > \max(A)$; otherwise, it is $\{\max(B) \dots \min(A)\}$. The combination of three sets A, B, and C, designated $\otimes(A, B, C)$, is the intersection of A, B, and C, if not empty; otherwise, the two sets among the A, B, and C that are furthest apart are found, and the range between them is intersected with the third set's range to yield the assignment. This can also be calculated as:

$$\otimes(\otimes(A, B), C) \cap \otimes(\otimes(A, C), B) \cap \otimes(\otimes(B, C), A).$$

Ordered

Assignment of downpass set to terminal nodes:

| State observed for terminal taxon | Downpass set assigned to terminal node | Steps counted |
|---------------------------------------|--|---------------------|
| One observed state, t | {t} | 0 |
| Observed set of states T, uncertain | range(T) | 0 |
| Observed set of states T, polymorphic | range(T) | $\max(T) - \min(T)$ |
| Missing data | {0...maxstate} | 0 |

"Maxstate" means the maximum observed or allowed state in the character. (This is not the same as the MAXSTATE option referred to elsewhere in the manual.)

Assignment of downpass set to internal node N based on sets at descendant nodes:

When N has two descendants P and Q, with downpass sets D_P and D_Q : Assign $\otimes(D_P, D_Q)$. Steps: Count the distance between D_P and D_Q if they do not overlap.

When N is polytomous, with k descendants with downpass sets $D_1 \dots D_k$:

"Soft" polytomy: Assign intersection of $D_1 \dots D_k$, if not empty; otherwise, assign the closed interval bounded by the smallest maximum to the largest minimum among the $D_1 \dots D_k$. Steps: Count the number of steps between the smallest maximum and largest minimum if the intersection of $D_1 \dots D_k$ was empty.

"Hard" polytomy: Assign intersection of $D_1 \dots D_k$, if not empty; otherwise, assign the set of all those states s such that $R_s \leq L_{s+1}$ and $L_s \leq R_{s-1}$, where R_s is the number of sets among the $D_1 \dots D_k$ all of whose elements are greater than s, L_s is the number of sets all of whose elements are less than s. Steps: Choose one such state s and sum the distances from s to each of the sets whose elements do not include s.

When N is an observed taxon fixed as an ancestor, with descendant with downpass set D_P :

N's observed set of states T is polymorphic or monomorphic: Assign range(T).

N's observed set of states T is uncertain: Assign $\otimes(\text{range}(T), D_P)$.

N has missing data: Assign D_P .

Steps: Except if N has missing data, count the distance between $\text{range}(T)$ and D_p .

Assignment of uppass set to N:

See description under "Unordered characters".

Assignment of final MPR set to terminal node N:

Define B to be N's uppass set if N's ancestor is a soft polytomy; otherwise, B is final MPR set at N's ancestor.

When N was observed monomorphic: Assign observed state.

When N was observed polymorphic, or with uncertain states: Assign $\Delta(D_N, B)$.

When N had missing data: Assign B.

Assignment of final MPR set to internal node N based on sets at surrounding nodes:

When N is the root: Assign N's downpass set to be the final MPR set.

When N has uppass set U_N and two descendants P and Q with downpass sets D_P and D_Q : Assign $\otimes(D_P, D_Q, U_N)$.

When N has uppass set U_N , downpass set D_N and is polytomous, with k descendants with downpass sets $D_1 \dots D_k$:

"Soft" polytomy: Assign $\otimes(D_N, U_N)$. (Recall node is below polytomy; see [page 348](#).)

"Hard" polytomy: Use the same calculations as for hard polytomies in downpass, except use sets $D_1 \dots D_k$ and U_N .

When N is an observed taxon fixed as an ancestor, with uppass set U_N , and with descendant P with downpass set D_P :

When N was observed monomorphic or polymorphic with observed states T: Assign $\text{range}(T)$.

When N had uncertain states, set T: Assign $\Delta(\text{range}(T) \otimes (\text{range}(T), D_P, U_N))$.

When N had missing data: Assign $\otimes(D_P, U_N)$.

Irreversible characters

Camin and Sokal (1965) have discussed the use of irreversible characters. That part of the algorithm in MacClade that handles uncertainties and missing data is original to MacClade. Polytomies are not allowed for irreversible characters because of difficulties with uncertain states and soft polytomies.

As mentioned above, the algorithms combine state sets of nearby nodes. For irreversible characters, designate the combining operator by \diamond . It is assumed that each state set contains all states between any two of its states; that is, there are no gaps in the set. (If there are, convert the set to its own range.) The combination of the sets A and B of two descendant nodes with the final MPR set C at an ancestral node, designated $\diamond(A, B, C)$, is the union of the [intersection of A, B, and C] with the [intersection of the (smaller of the minimum values in A and B) with the (range from minimum value in C to the larger of the minimum values in A and B)]. This can be written $(A \cap B \cap C) \cup ([\min(\{\min(A), \min(B)\})] \cap \{\min(C) \dots \max(\{\min(A), \min(B)\})\})$.

Irreversible

Assignment of downpass set to terminal nodes:

| State observed for terminal taxon | Downpass set assigned to terminal node | Steps counted |
|---------------------------------------|--|---------------------|
| One observed state, t | {t} | 0 |
| Observed set of states T, uncertain | $\text{range}(T)$ | 0 |
| Observed set of states T, polymorphic | $\{\min(T)\}$ | $\max(T) - \min(T)$ |
| Missing data | $\{0 \dots \text{maxstate}\}$ | 0 |

"Maxstate" means the maximum observed or allowed state in the character. (This is not the same as the MAXSTATE option referred to elsewhere in the manual.)

Assignment of downpass set to internal node N based on sets at descendant nodes:

N has two descendants P and Q, with downpass sets D_P and D_Q : Assign intersection of the D_P and D_Q , if not empty;

otherwise, assign set consisting of the state that is the smaller of the maximum values of D_p and D_q . Steps: If intersection was empty, count steps from smaller maximum to larger minimum.

N is an observed taxon fixed as an ancestor, with descendant with downpass set D_p :

- When N 's observed set of states T is polymorphic or monomorphic: Assign $\text{range}(T)$.
- When N 's observed set of states T is uncertain: $\triangleright(\text{range}(T)D_p)$.
- When N has missing data: Assign D_p .
- Steps: Except if N has missing data, count the distance between $\text{range}(T)$ and D_p .

Assignment of uppass set to N :

No uppass sets are assigned.

Assignment of final MPR set to terminal node N with downpass set D_N and ancestor M with final MPR set F_M :

- When terminal taxon was observed monomorphic or polymorphic: Assign D_N .
- When terminal taxon was observed with uncertain states: Assign $\triangleright(D_N F_M)$.
- When terminal taxon had missing data: Assign F_M .

Assignment of final MPR set to internal node N based on sets at surrounding nodes:

- When N is the root: Assign N 's downpass set to be the final MPR set.
- When N has two descendants P and Q with downpass sets D_p and D_q , and ancestor M with final MPR set F_M : Assign $\triangleright(D_p, D_q, F_M)$.
- When N is an observed taxon fixed as ancestor, with set T of observed states and with descendant P with downpass set D_p , and ancestor M with final MPR set F_M :

 - When T is monomorphic or polymorphic: Assign T .
 - When T has uncertain states: Assign $\text{range}(T) \cap \triangleright(D_p, \text{range}(T), F_M)$.
 - When T is missing data: Assign $\triangleright(D_p, \{0 \dots \text{maxstate}\}, F_M)$.
 - If the result is empty, irreversibility has been violated by the ancestral placement.

Stratigraphic characters

MacClade's implementation of Fisher's (1992) stratigraphic parsimony uses algorithms developed by W. Maddison and described below. *These algorithms are complex, and though they have been intensively examined and tested, they are considered unstable and in certain circumstances may not be completely valid. To document the algorithms' behavior, a test data file showing numerous examples is included with MacClade.* These algorithms are modified from the algorithms for irreversible characters. Users are advised to read the description below so as to be able to interpret better the somewhat unorthodox output MacClade gives with stratigraphic characters. This is especially important because MacClade's tracing of stratigraphic characters does not indicate ages of ancestors in the same way that its tracing of other characters indicates states of ancestors.

Algorithms

The basic rules for assigning ages to ancestral nodes are as follows: (1) ancestors must predate descendants; (2) ancestors should be assigned ages so as to minimize implications of missing fossils.

Stratigraphic

Assignment of downpass sets, final MPR sets for all cases except when observed taxa fixed as ancestors:

Use the same algorithms as for irreversible characters, but count steps only as indicated below.

When observed taxa are fixed as ancestors:

Assignment of downpass sets to internal nodes: Treat the tree as if the observed taxa were not fixed as ancestors and use the irreversible character algorithms above.

Assignment of final MPR set to root node N when N is an observed taxon fixed as ancestor, with set T of observed states and downpass set D_N , and with descendant node P with downpass set D_p :

When N is polymorphic (i.e., observed in more than one stratum): Assign $\triangleright(\text{range}(T), D_p, \text{range}(D_N \cup \{\text{max}(D_p)\}))$.

When N is not polymorphic: Assign D_N .

Assignment of final MPR set to internal node N (not the root) when N is an observed taxon fixed as ancestor, with set T of observed states, and with descendent node P with downpass set D_P , and with ancestral node M with final MPR set F_M :

When T is monomorphic: Assign T.

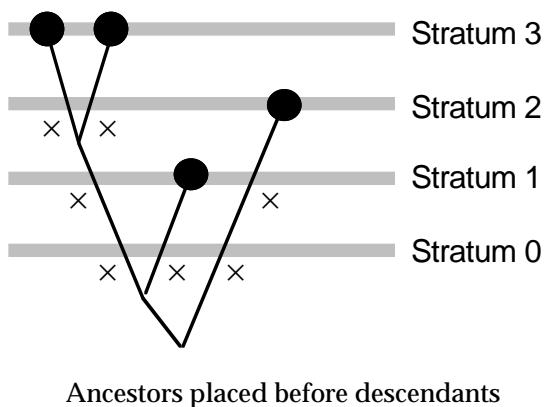
When T is missing data: Assign $[F_M \cap D_P] \cup [\min(F_M) \dots \min(D_P)]$.

When T is uncertainty: Assign the state H just below the first gap in T — if H is larger than $\min(F_M)$, assign $\triangleright(\text{range}(T), D_P, \text{range}(F_M \cup \{\min(H, \max(D_P))\}))$, otherwise $\triangleright(\text{range}(T), D_P, F_M)$.

When T is polymorphic: Assign $\triangleright(\text{range}(T), D_P, \text{range}(F_M \cup \{\max(D_P)\}))$.

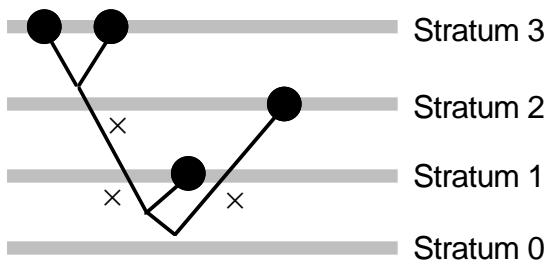
Explanation of algorithm

Suppose we had a tree of four taxa occurring in various strata, and we drew the branches of the tree so as to indicate possible ages of the various parts of the tree. If at first we only constrained our placement of ancestors so that they were older than descendants, we might end up with something like that shown below.



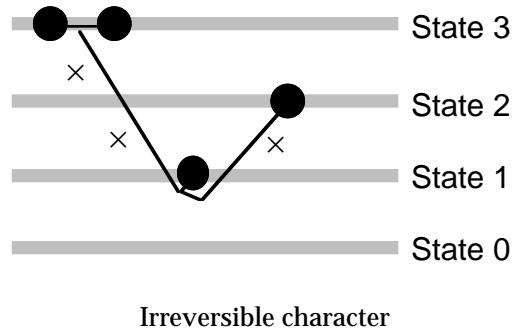
Ancestors placed before descendants

The x's mark instances where a lineage crosses a stratum without represented fossils. Clearly this reconstruction has more such absences than necessary. There is no point in having two adjacent lineages crossing the same stratum if by moving their common ancestor above the stratum only one crossing would be required. The most parsimonious assignment of ages to ancestors would therefore push all branch points as high a possible, so that the number of lineages crossing strata would be minimized, as below.



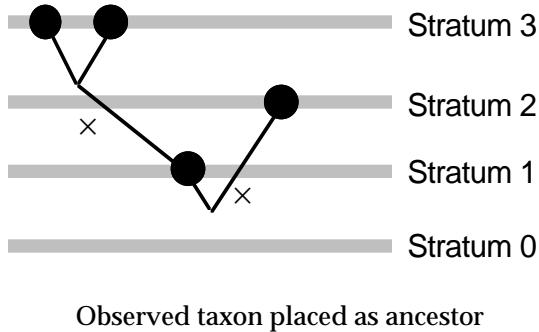
Minimizing stratum crossings

The rule is therefore that one pushes the ancestors as high as possible without violating the principle that ancestors must predate descendants. The rule for irreversible characters is the same except that ancestors and descendants are allowed to coexist on the same state. Thus an irreversible character would have its ancestral states reconstructed as shown below.



In this figure the x's represent not the crossing of strata but changes from one state to the next. The number of x's is the same as the number of stratum crossings counted above because a stratum crossing is required every time a descendant is assigned to a younger stratum and there is not an observed ancestor to represent this crossing, just as a character state transition in an irreversible character is required every time a descendant is assigned to a higher character state number than its ancestor. Thus, the irreversible algorithm can be used without modification to count the minimal number of stratum crossings and reconstruct stratigraphic ages when none of the observed taxa are treated as ancestors. However, an assignment of stratum i to an unobserved ancestral branch point must be interpreted as saying that the branch point was just below the stratum, to avoid the prohibition against coexistence of ancestors and descendants. This is a point worthy of note. When you see that MacClade assigns stratum i to an ancestral node, do not take this as indicating that MacClade reconstructs the ancestor to have occurred exactly in stratum i , unless the ancestor was an observed taxon considered as a direct ancestor.

If there are observed taxa that are considered direct ancestors, then the number of character transitions that would be counted if age were simply considered an irreversible character cannot be directly used to indicate the number of stratum crossings. The reason for this is that the descent of a lineage from one stratum to the next younger would not have to require an unrepresented crossing of the older stratum if there were an observed ancestor to represent the crossing:



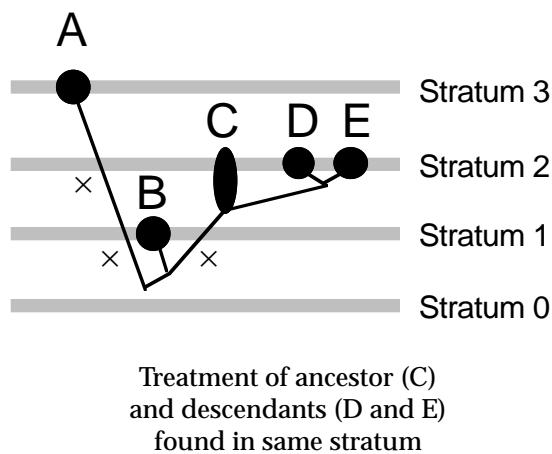
With irreversible characters, a state transition would have to be recorded regardless of whether the ancestor was observed or not. This difference can be easily accounted for by modifying the irreversible algorithm so that whenever a taxon is considered a direct ancestor, it can be used to account for a stratum crossing without cost.

MacClade therefore uses the irreversible algorithm with several modifications in order to reconstruct ancestral stratigraphic ages. The modifications concern observed taxa fixed as ancestors. For stratigraphic characters, the state at a node that is an observed taxon fixed as ancestor is reconstructed as if the observed taxon were still terminal. That is, this node is not forced to take the observed stratigraphic ages of the

taxon. For instance, if you place an observed taxon known only from higher strata in ancestral position to taxa known from lower strata, MacClade will suppose the observed taxon actually extended deeper, and the actual ancestors were organisms with the exact same states in all nonstratigraphic characters but in lower strata. This is done to avoid ancestors being younger than descendants.

In the above discussion it was noted that an ancestor and its descendant could not coexist in the same stratum. MacClade enforces this, although there are circumstances in which coexistence appears to be allowed.

For instance, suppose species C, D, and E were all found in stratum 2, yet C was designated as being ancestral to D and E, as shown below. In this case, MacClade interprets the designation of C as an ancestor as not applying only to the members of C sampled from stratum 2, but rather to identical forms just below stratum 2, in the inter-stratum zone between 1 and 2. These ancestral forms gave rise to D and E. This allows MacClade to avoid supposing that an ancestor and its descendants coexisted in stratum 2.



In MacClade's tracing of a stratigraphic character, a branch is shaded with a special pattern (▨▨▨▨) if it is reconstructed to have multiple states. With other character types, an observed taxon fixed as an ancestor is shaded by the polymorphic or uncertain pattern depending on whether the taxon was coded with the AND or the OR separator (see [Chapter 13](#)). With stratigraphic types, it is difficult to determine whether multiple states reconstructed at a node represent uncertainty in the node's position or the spanning of several strata by the node. MacClade therefore ambiguously indicates multiple states with the special pattern.

Counting stratigraphic parsimony

The following description gives an outline of MacClade's algorithm to count steps in stratigraphic characters. The original source code should be consulted for full details.

To count stratigraphic parsimony, the total cost of implied but unobserved crossing of strata (missing fossils), MacClade reconstructs ancestral stratigraphic positions then counts implied stratum crossings. After the ancestral states in the stratigraphic character are reconstructed, MacClade begins at the root and finds one resolution of any ambiguity in stratigraphic position. The state so resolved for node N will be called S_N . Each node N is assigned the highest state in its final MPR set, to take maximal advantage of its accounting for crossing strata. As the counting algorithm moves up the tree, it treats each node N as follows. Assume the node ancestral to N is M.

Designating resolved stratum assignments:

When N is an internal node, but not an observed taxon fixed as ancestor:

When N is root: Designate S_N to be $\max(F_N)$.

When N is not root: Designate S_N to be the larger of S_M and $\min(F_N)$.

When N is an observed taxon:

When N is observed taxon fixed as ancestor: Designate S_N to be $\max(F_N)$.

When N is observed terminal taxon: Designation not needed.

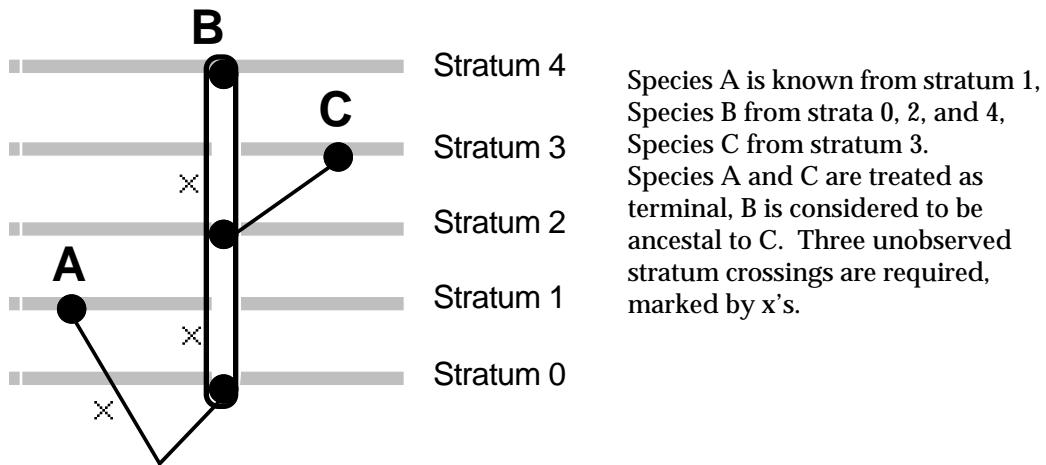
Counting steps (these alternatives are not mutually exclusive):

When N is an internal or external node, and is not the root: If S_M is below the lowest stratum in F_N ($\min(F_N)$), then this implies that N's ancestral lineage crossed the intervening strata. Count the stratum crossings from S_M to $\min(F_N)$, not including the crossing of the stratum $\min(F_N)$, but including the crossing of S_M if its crossing was not accounted for earlier in the tree. Record that S_M 's crossing is now accounted for, but that $\min(F_N)$'s crossing has not been accounted for.

When N is observed taxon fixed as ancestor: Count unobserved stratum crossings for any strata from S_M to S_N that are not included among N's observed strata (if the lowest observed stratum of N is above S_N , then count stratum crossings from S_N to this lowest). S_N , which was designated as the highest stratum in F_N , will be passed to any nodes descendant from N, to which it will be known as S_M (see above). When this value is passed up, it will be indicated that this stratum's crossing has been accounted for if S_N is among N's observed or possibly observed strata, unless its descendent node is reconstructed as having this same state, in which case the stratum cannot be considered accounted for without violating the prohibition against co-occurrence of ancestors and descendants in the same stratum.

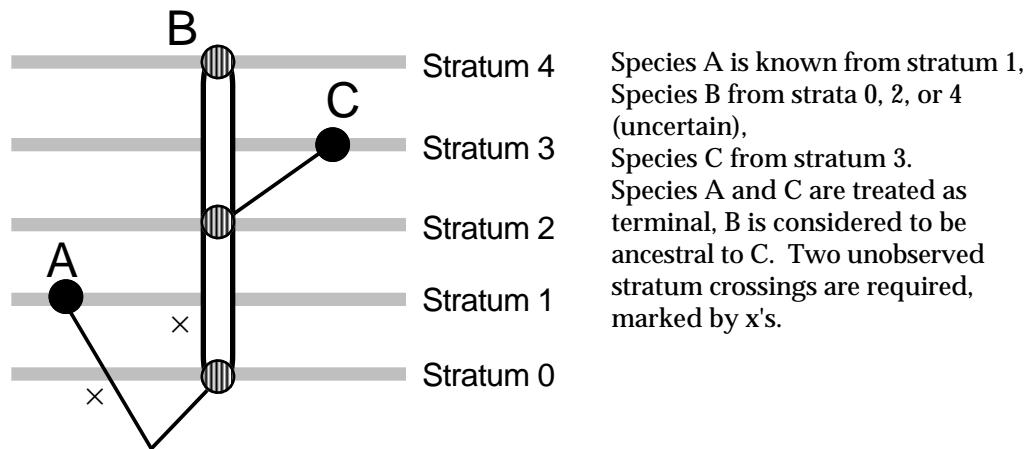
When N is polymorphic observed taxon (i.e., known from more than one stratum): Count unobserved stratum crossings for any strata not included within the taxon's observed strata but that are between the lowest and highest observed crossing.

The most complex part of the counting concerns nodes that are observed taxa placed in ancestral position. If N is an observed taxon fixed as an ancestor, then the strata in which it has been observed are therefore accounted for by observed fossils, and we need not count unobserved stratum crossings. Therefore, where possible, the observed fossils are placed along the branch leading up to the taxon's descendant node. If we have a species B known from strata 0, 2, and 4, and its descendant species C is known from stratum 3, then those samples of species B from strata 0 and 2 can be assumed to be on the line directly ancestral to species C:

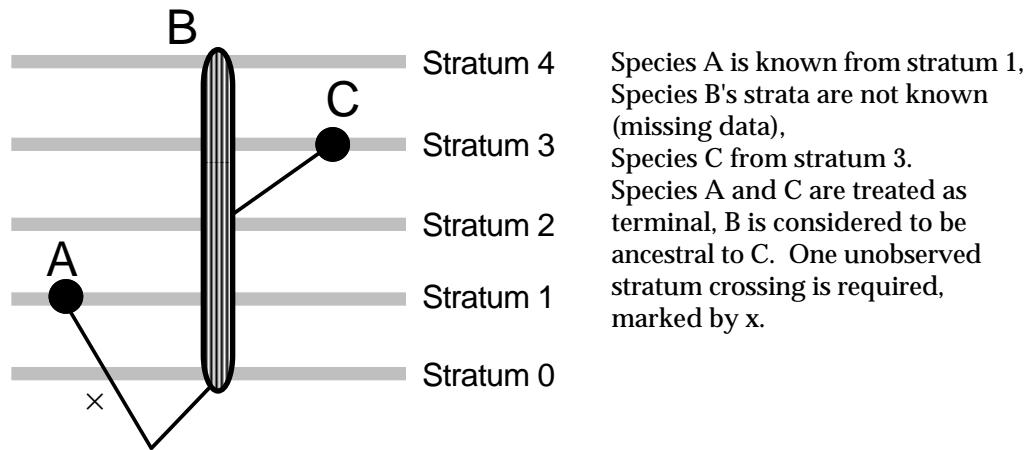


Note that in this example the absence of stratum 3 from samples of species B is taken to require an extra stratum crossing.

If species B's stratigraphic position is uncertain, then the algorithm assumes optimistically (for parsimony's sake) that B may be known from each of strata from which samples might possibly derive:



If species B's stratigraphic position is completely unknown (missing data), then the algorithm assumes optimistically that B may be known from all strata:



This convention for dealing with uncertain stratigraphic position leads to some difficulties. If many fossil samples are known but none are well dated or correlated, then it may be reasonable to assume optimistically that all strata are represented among the samples. However, if only one specimen is known, but its stratum is not well dated, then clearly we cannot hope that all strata are represented; at most only one stratum is represented by this sample. MacClade follows its convention because it does not yet have the means to indicate in its data matrix both what states a taxon might have in an uncertain character as well as how many states it might have (i.e., "Taxon has exactly one of states 0, 2, or 4", versus "Taxon has one or more of states 0, 2, or 4").

Giving different weights to absences in different strata is easily incorporated into the algorithm, because the need to push ancestors as high as possible without violating ancestor precedence does not depend on these different weights. Thus the stratum crossings required are reconstructed the same way regardless of which weights are assigned (unless they are assigned 0), and the only difference is how much we count for each of the crossings so reconstructed.

Dollo characters

A discussion relevant to reconstructing ancestral states for two-state Dollo characters was presented by Farris (1977). The algorithm in MacClade allows multistate Dollo characters. That part of the algorithm in MacClade that handles missing and uncertain data is original to MacClade.

MacClade performs Additive Binary Recoding (Sokal and Sneath, 1963) on multistate Dollo characters. What is described below is the binary version of the algorithm. At each node N is stored two sets: the set recording the last decisive split (E_N), and the set recording the states that would have been assigned to that node were it the root and the clade above it the entire tree (A_N). The E_N set can be thought of as recording what states were on either side of the nearest descendant of N that had nonmissing data in both left and right descendant clades. MacClade's algorithms allow only dichotomous trees with all observed taxa terminal.

Dollo

Assignment of downpass sets E_N and A_N to terminal node N:

| State observed for terminal taxon | E_N | A_N |
|--------------------------------------|-------|-------|
| One observed state, 0 | {0} | {0} |
| One observed state, 1 | {1} | {1} |
| States 0 or 1, uncertain, or missing | {} | {0,1} |
| States 0 and 1, polymorphic | {0,1} | {0,1} |

Assignment of downpass sets E_N and A_N to internal node N based on sets at descendent nodes:

| Given E_P and E_Q of | | On downpass, assign N | |
|--------------------------|-------|-----------------------|-------|
| E_P | E_Q | E_N | A_N |
| {0} | {0} | {0} | {0} |
| {0} | {0,1} | {0,1} | {0} |
| {0} | {1} | {0,1} | {0,1} |
| {0} | {} | {0} | {0} |
| {0,1} | {0,1} | {1} | {1} |
| {0,1} | {1} | {1} | {1} |
| {0,1} | {} | {0,1} | A_P |
| {1} | {1} | {1} | {1} |
| {1} | {} | {1} | {1} |
| {} | {} | {} | {} |

Assignment of uppass set to N:

No uppass sets are assigned.

Assignment of final MPR set to internal node N based on sets at surrounding nodes:

Once the root is reached on the downpass, the set A_R assigned to the root is the most parsimonious set based on the whole tree, that is, it is the final MPR set for R, F_R . The algorithm then can proceed up the tree assigning final MPR sets to all nodes using the E sets of descendants and the final MPR sets of ancestors, as shown below. Assume N has two descendants, P and Q, and ancestor M.

| E sets of descendants | | Final pass assignments to N for various values of FM (final MPR sets at M) | | |
|-----------------------|-------|---|--------|----------|
| EP | EQ | FM={0} | FM={1} | FM={0,1} |
| {0} | {0} | {0} | {0} | {0} |
| {0} | {0,1} | {0} | {1} | {0,1} |
| {0} | {1} | {0} | {1} | {0,1} |
| {0} | {} | {0} | {0,1} | {0,1} |
| {0,1} | {0,1} | {1} | {1} | {1} |
| {0,1} | {1} | {1} | {1} | {1} |
| {0,1} | {} | {0} | {1} | {0,1} |

| | | | | |
|-----|-----|-------|-----|-------|
| {1} | {1} | {1} | {1} | {1} |
| {1} | {} | {0,1} | {1} | {0,1} |
| {} | {} | {0} | {1} | {0,1} |

Once F_N has been calculated for each of the binary factors, they can be recomposed into the final MPR set at node N for the original multistate character.

Assignment of final MPR set (not binary recoded) to terminal node N:

Assume that the binary recoding has been reversed to obtain the multistate final MPR set at N's ancestor. Define B to be final MPR set at N's ancestor.

When terminal taxon was observed monomorphic: Assign observed state.

When terminal taxon was observed polymorphic or with uncertain states: Assign $\Delta(\text{range}(\text{observed states}), B)$.

When terminal taxon had missing data: Assign B.

Counting steps: No simple method has been described to count steps in a single pass through the tree; MacClade fully reconstructs the ancestral states, then counts steps in the reconstruction.

Step matrices and character state trees

Algorithms for finding the most parsimonious ancestral states when symmetrical step matrices are used were originally described by Sankoff and Rousseau (1975) and Sankoff and Cedergren (1983). In MacClade 2.1 (Maddison and Maddison, 1987) we extended their treatment by allowing asymmetries in distance for gains and losses. Williams and Fitch (1989, 1990) have redescribed the step-matrix algorithm.

In MacClade 4, character state trees are first converted into step matrices and the same algorithms are used as for step-matrix types. This is much less efficient than an alternative method, namely to use additive recoding to break the character state tree into ordered characters.

The algorithm for step-matrix characters follows a similar path through the tree as those for the other character types described above, but differs in that assigned to each node is not a set of states, but rather a number for each possible state. This number records how many steps are required in a section of the tree represented by the node, given that the particular state is assigned to the node. On the downpass, the number records how many steps are required in the clade above the node; on the uppass the number records how many steps are required in the area of the tree below and beside the node. Thus for each possible state that could be assigned to a node, there is a graded scale of goodness, in contrast to the other algorithms in which a state is either included or excluded in a state set as a viable candidate. The algorithm's flexibility, as well as its slowness and other limitations, are due to its keeping records on a graded scale.

The algorithm likewise makes a downpass and an uppass, so that after these passes one has recorded the opinions from the parts of the tree represented by a node's left and right descendants and its ancestor. On the downpass the record for a node N is calculated as follows. Recall that the information to be recorded for N consists of (for each possible state) the number of steps required in N's clade, given that N is assigned this state. So, the algorithm tries assigning each possible state s to N, and for each the minimum number of steps on the left side (N to its left descendant node P and above) and right side (N to its right descendant node Q and above) are added to yield the number of steps required in total in N's clade. The minimum number of steps on the left side, for instance, is calculated by finding the state t assigned to P that yields the lowest sum of steps between N (having s) and P (having t) plus the number of steps required in the clade of P, given P is assigned t. This latter number of steps is available from the information already calculated for P. Thus the records of the two nodes P and Q are combined to yield the information for N. This basic pattern of calculation is followed on the uppass and in the final decision for any given node in which the information for three nodes is combined. After this is done, the states actually assigned to a node are those that require the minimum number of steps according to the whole tree. The minimum number of steps required by the character is directly obtained as part of the reconstruction pro-

cess: it can be found in the information stored at the node after its final states are reconstructed.

The step-matrix algorithm is therefore conceptually simple. Only one aspect needs particular attention: What to do in terminal taxa that are polymorphic? The following assignments are made:

| Assignment of <u>downpass</u> array to terminal node N: | |
|---|---|
| Missing | 0 cost for all states. |
| Monomorphic | 0 cost for observed state, infinity for all others. |
| Uncertain | 0 cost for possible observed states, infinity for all others. |
| Polymorphic | |
| Two states, b and c | If $a=b$, then cost of state a is cost of change a to c. If $a=c$, then cost of state a is cost of change a to b. Otherwise, then cost of state a is minimum cost among the three possibilities for changes, $a \rightarrow b \rightarrow c$, $a \rightarrow c \rightarrow b$, and $b \leftarrow a \rightarrow c$. |
| Three or more states | If a is one of the observed states, then cost of state a is minimum cost of a change from a to any other state in polymorphism. otherwise, cost of state a is infinity. |

For missing data, the assignment allows any states to be placed at the terminal node with no cost within the terminal taxon. For taxa that are monomorphic or multistate uncertain, then any of the possible states are allowed at no cost but unobserved states are prohibited. For taxa that are polymorphic with two states, the above assignment presumes that states b and c might be evolved within the terminal taxon, and it accurately assesses the minimum number of steps required, given state a is placed at the terminal node.

For taxa polymorphic with three or more states, it is very difficult to assess accurately what changes are required within the terminal taxon. For this reason, the assignment indicated above is at best only an approximation. The best approach would be to find the length of the most parsimonious tree linking all the states observed in the terminal taxon and rooted at state a. This calculation, however, would be involved and slow; MacClade does not do it. Although MacClade might simply ignore steps within these polymorphic taxa and treat them as if they were uncertainties (as PAUP* does), instead it assigns steps for putting a state ancestral in the taxon according to the distance to just the single closest other state also observed in the taxon. MacClade does this to avoid assigning states that would imply impossible transitions. If MacClade encounters a terminal taxon polymorphic for three or more states in a user-defined character, it will warn you.

Polymorphic terminal taxa are particularly problematic when the step matrix in use violates the triangle inequality. As noted on [page 72](#), a matrix violating the triangle inequality might be assumed to be describing the cost for state-to-state transformations during one branch length's worth of time. If we are allowing changes within terminal taxa, as MacClade does when taxa are polymorphic, then there are no explicit branches within terminal taxa. Matrices violating the triangle inequality may therefore behave in strange and inconsistent ways.

All of these calculations use the user-input matrix of the distances from one state to another. When you put a " ∞ " in the matrix to indicate the change is prohibited, MacClade treats the distance as infinity (in fact, it stores it as -1). If MacClade indicates that a character requires an infinite number of steps on the tree, then you must have created impossible assumptions, for instance by constructing a step matrix prohibiting any changes.

Fixing states

When the state at a node is fixed to a state of the user's choice in the character traced (by the paintbrush tool), then MacClade forces the downpass, uppass, and finalpass sets at that node to consist of just the

fixed state.

If the ancestor of a node N has its state fixed and N's ancestor is either dichotomous or a hard polytomy, then N's uppass set is no longer calculated from the downpass sets of its sister(s) and the uppass set of its ancestors, because the fixing of state at the ancestor effectively insulates N from the effects of its sisters. The uppass set of N is taken from just the state fixed at the ancestor. On the other hand, for soft polytomies, because the ancestral node fixed is treated as just below the basal node of any resolution of the polytomy, the sister's effects are not insulated, and the uppass set at N is calculated from the sisters' downpass sets and the fixed state at the ancestral node.

Because the normal downpass calculations are suspended when a node with fixed state is encountered, the number of steps required locally must be calculated specially. For unordered, ordered, and irreversible characters the fixed state is compared to the downpass sets at each of the descendant nodes. If N is dichotomous or hard polytymous, then the distance from N's fixed state to each of the descendant downpass sets is measured. If N is soft polytymous, then N is actually just below the basal node of any resolution of the polytomy, and the distance need be measured only from N to the downpass set that would have been assigned to the basal node. If N is a terminal taxon, then recall that fixing N's state is assumed to fix a state just below the terminal taxon (see [Chapter 18](#)). Steps must be counted from the fixed states to the downpass set that would have been assigned to the terminal node.

For user-defined (step-matrix) characters, the downpass information at the node is adjusted so that the cost for assigning the fixed state to the node is calculated as if the node were not fixed (using information from the descendent nodes), whereas the cost for assigning any other state is infinite.

Assumptions such as irreversibility or the Dollo assumption can be violated by fixing the states at nodes. For irreversible characters MacClade will warn you that irreversibility has been violated. If irreversibility is violated, the character tracing treats the character like an ordered, reversible character in the region of the violation. For user-defined types infinite numbers of steps may be required when you make an illegal fixing; MacClade will warn you if an infinite number of steps is required. For Dollo characters MacClade will give *no warning* that you have violated the Dollo assumption.

Continuous characters

The algorithms used for continuous-valued characters are those of Swofford and Maddison (1987) for linear parsimony and of W. Maddison (1991) for squared-change parsimony. The MINSTATE and MAXSTATE reconstructions are derived after a full linear-parsimony reconstruction by selecting either the minimum or maximum values in final MPR sets at the nodes, respectively.

The algorithms for squared-change parsimony in some respects resemble those for step matrices, in that the information stored at each node indicates how good is each state (W. Maddison, 1991; see McArdle and Rodrigo [1994] for an alternative using the algorithms of matrix manipulation instead of those of tree recursion). For instance, on the downpass a quadratic function is stored at each node that indicates, for each state that might be placed at the node, the summed squared length required in the clade above the node. It is calculated from the two quadratic functions already obtained for the two descendent nodes. A pass down the tree obtains the downpass quadratic functions at each node; a pass back up the tree obtains the uppass quadratic functions. A final pass combines quadratic functions from above and below each node to yield the quadratic function for the node. This final function summarizes the preferences of the whole tree for the states at the node. The most parsimonious single state at the node comes directly from this function (W. Maddison, 1991).

The linear parsimony reconstruction tends to allow change to concentrate on a few branches; the squared-change reconstruction forces changes to spread out more evenly over the tree. W. Maddison (1991) notes that the squared-change reconstruction can be considered a Bayesian estimate under a Brownian motion model of evolution (see also Schluter et al., 1997).

Equally parsimonious reconstructions

When there are several equally parsimonious reconstructions of evolution, as in the example in the section "Ancestral states and reconstructions", above, we might want to examine each of them. The initial character tracing that showed equivocal branches did not directly indicate to us that there were six reconstructions nor what they were. We knew only that there was more than one way to resolve the ambiguity. To find all of the fully resolved reconstructions of character evolution, special algorithms are needed to supplement the algorithms described above. Swofford and Maddison (1987) have discussed means by which to examine all the alternative reconstructions.

Algorithms have also been developed that present to the user one or a very few of the most parsimonious reconstructions. Some of these will be discussed after methods for discovery of all most-parsimonious reconstructions are outlined.

Examining all most-parsimonious reconstructions

A user might wish to examine each of the MPRs, to see how they differ in their implications about some aspect of character evolution. A user can examine all of the MPRs sequentially using MacClade's Show MPRs Mode, available in the Trace menu (this option had been named "Equivocal Cycling" in MacClade 3).

The first reconstruction found by this mode is that which places the lowest-valued parsimonious states at the nodes, giving priority to nodes closer to the root. That is, the algorithm first finds the nodes closest to the root that have equivocal assignments (perhaps the root itself). Each such node is assigned the lowest-valued state among its parsimonious assignments (the smallest state in its MPR set). Given these assignments, the algorithm may recalculate the parsimonious assignments to nodes further up, and then looks for the next nodes further up that have equivocal assignments. Each such node is assigned the lowest-valued state among its parsimonious assignments. Given these assignments, the algorithm then moves even higher, away from the root. In this way the algorithm stepwise assigns lowest-valued states, each assignment possibly depending on assignments made closer to the root. It should be understood that if the character is of unordered or step-matrix type, an assignment of a lower-valued state near the root might force assignments of higher-valued states further up the tree. Thus the first reconstruction does not necessarily place at each node the lowest state in its MPR set.

The last reconstruction found by the **Show MPRs Mode** is that which places the highest-valued parsimonious states at the nodes, giving priority to nodes closer to the root. Because assignments to lower nodes affect assignments to higher nodes, the last reconstruction does not necessarily place at each node the highest state in its MPR set.

Between the first and last reconstruction, this mode moves through alternative reconstructions in a particular sequence. The Show MPRs Mode moves through higher and higher reconstructions by raising the states at each branch, with the leftmost branches changing the fastest (like the 1s digit in a counter), and the rightmost branches changing the slowest (like the 1000s digit in a counter). The raising of states is done subject to the constraint that only parsimonious reconstructions are made. Eventually, one arrives at the last reconstruction.

In MacClade 3, the same mode was used in various of MacClade's calculations, such as for Tree Changes, Trace All Changes (minimum-average-maximum) and for the State Changes & Stasis chart (average), which require all equally parsimonious reconstructions to be examined. For these calculations, MacClade would automatically build each MPR, and examine it to see the number and distribution of changes on the tree. If many MPRs exist, perhaps millions or more, stepping through each MPR is a time-consuming process. For this reason new algorithms were built into MacClade 4 that allow much quicker calculations when there are many MPRs, as described in the next section.

Examination of all MPRs during MacClade's calculations

For those calculations in which either extreme values or averages across MPRs or the number of MPRs need to be calculated, all MPRs need to be explicitly or implicitly examined. MacClade 3 explicitly examined each MPR (a mode still available in MacClade 4, labeled "explicitly build and examine each MPR", available via the **MPR Calculations Mode** menu item under **Trace**). However, this mode can be extremely slow when many MPRs are present, and so new algorithms have been added to MacClade 4, in which all MPRs are only implicitly examined.

The algorithms that implicitly examine all MPRs follow the principles of Rinsma et al. (1990), and thus are in the style of those described for "[Step matrices and character state trees](#)" on page 93, and those of Sankoff and Rousseau (1975) and Sankoff and Cedergren (1983). These algorithms are described here:

Number of MPRs

During a downpass, a number ($M_{N,i}$) is stored at each node N for each state i . This records the number of MPRs present in the subtree above the node, given that state i is assigned to the node. Define $F_{L|N \leftarrow i}$ to be the MPR set for the left descendant node given assignment of state i to node N. $F_{L|N \leftarrow i}$ is calculated using procedures described above in the sections on finding MPR sets, and the number of MPRs above the left descendant node for each of the elements in $F_{L|N \leftarrow i}$ are summed; this is multiplied by the equivalent value for the right descendant, to yield the value for node N for that state. When the downpass reaches the root (R) of the tree, one can simply sum $M_{R,j}$ for all states j in the MPR set for the root to yield the total number of MPRs.

Number of changes on the tree

In calculating the number of changes on the tree (which you can request by turning on **Tree Changes** in the Σ menu), MacClade stores in the downpass two numbers for each state in the MPR set at each node: the minimum number of changes that would be present above that node if that state were assigned to the node, and the maximum number of changes. Let the minimum number of changes stored for state i at a node N be $c_{N,i}$. $c_{N,i}$ is calculated as the minimum value, for all $j \in F_{L|N \leftarrow i}$, of

$$\begin{cases} 1+c_{L,j} & \text{if } i \neq j, \\ c_{L,j} & \text{if } i = j \end{cases}$$

summed with the equivalent minimum for the right descendant. After the downpass is completed, the minimum of the values for the states in the root's MPR set is the minimum number of changes in the entire tree. A similar calculation yields the maximum number of changes.

Minimum/maximum number of changes on a branch

The minimum or maximum number of changes over MPRs along the branch of node N (e.g., as determined for minimum-average-maximum mode in **Trace All Changes**) can be calculated by summing the minimum and maximum values for all characters. The calculation of the minimum value for a character is relatively simple. If the branch's ancestral node is called A, and then if one examines all states $i \in F_A$ (A's MPR set), and for each of these calculates $F_{N|A \leftarrow i}$ (the MPR set that would be at N if A is fixed to i), then calculates the minimum number of changes implied in changing from i to each of the states in $F_{N|A \leftarrow i}$ for all i , the minimum for the branch can be determined. A similar calculation yields the maximum number.

Average number of changes on a branch

The calculation for the average number of changes along a branch across MPRs is more complex. Let T be the total number of MPRs for the character. For a node N with ancestor A, the average (across MPRs) number of changes along the branch below N is the sum, over all possible state changes $j \rightarrow i$ (with $j \neq i$) of the fraction of MPRs exhibiting that state change along the branch from A to N. The fraction can be obtained by calculating the total number of MPRs exhibiting that change and dividing by T. To calculate the number

of MPRs that exhibit a $j \rightarrow i$ change along the branch, we begin with the downpass to calculate the number of MPRs that would be present above a node N if that node were assigned state i (let that value be called $M_{N,i}$). An uppass is also performed to calculate the number of MPRs that would be present *below* the node A if that node were assigned state j (let that value be called $B_{A,j}$). We also need to calculate the number of MPRs in the clade of N's sister (which we will call "S"), and for this we sum $M_{S,k}$ for all states k that are members of $F_{S|A \leftarrow j}$ (the MPR set that would be at S if A is fixed to j). The total number of MPRs exhibiting this change is thus $\left(\sum_{k \in F_{S|A \leftarrow j}} M_{S,k} \right) \cdot B_{A,j} \cdot M_{N,i}$. Calculations around the root of the tree differ slightly from this. The downpass calculations proceed through the tree combining M's of sister nodes; the uppass calculations combine B's of the ancestor with M's of the sister to calculate the B's for a node.

Minimum/maximum number of particular changes in tree

Calculations for the minimum and maximum number of changes of each sort as displayed in the **State Changes and Stasis** chart are similar to those for the total number of changes, except that rather than storing a value for the minimum and maximum number of changes above the branch should that node be assigned state i , two entire matrices are stored, one for the minimum number for each type of state change, and one for the maximum.

Average number of particular changes in tree

For the average number of changes of each sort as displayed in the **State Changes and Stasis** chart, MacClade could use calculations similar to that for the average number of changes along a branch, but, for historical reasons, it does not. MacClade performs a downpass calculating the number of MPRs that would be present above a node N if that node were assigned state p ($M_{N,p}$). Another downpass is conducted in which, for each state p at each node, a matrix is calculated of the average number of changes of each sort above that node given the node is assigned state p . The average number of $j \rightarrow i$ changes above node N ($\bar{C}_{N,p,j \rightarrow i}$) is calculated as the sum of the averages of the left descendent subtree plus the right descendent subtree. For example, consider the left descendent subtree. The MPR set at the left descendent node, L, given N is assigned state p ($F_{L|N \leftarrow p}$) is calculated. For each $k \in F_{L|N \leftarrow p}$ we have the average number of $j \rightarrow i$ above L given k at L, $\bar{C}_{L,k,j \rightarrow i}$. The average number of $j \rightarrow i$ changes above N if N is assigned p and L is assigned k is thus $\bar{C}_{L,k,j \rightarrow i}$ plus 1 if $j=p$ and $i=k$ (to account for the $p \rightarrow k$ change along the N to L branch). To sum the averages for all $k \in F_{L|N \leftarrow p}$ we need to weight the changes by the fraction of MPRs above L that have k assigned to L. The fraction of MPRs above node L that have state k assigned at node L ($\phi_{L|L \leftarrow k}$) is cal-

culated as $M_{L,k} / \left(\sum_{k \in F_{L|N \leftarrow p}} M_{L,k} \right)$. This fraction indicates the relative contribution of changes of MPRs

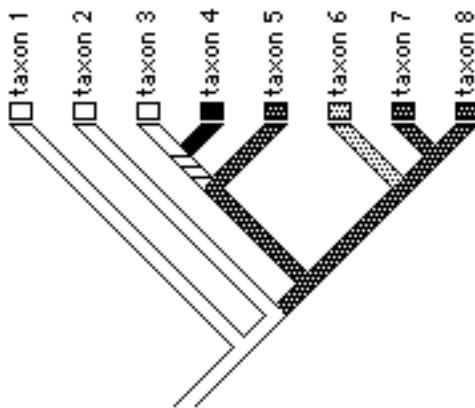
above L with k at L to the averages contributed by the L branch. $\bar{C}_{N,p,j \rightarrow i}$ is thus the sum for all $k \in F_{L|N \leftarrow p}$ of $\phi_{L|L \leftarrow k} \cdot (\bar{C}_{L,k,j \rightarrow i} + \beta)$, where $\beta=1$ if $j=p$ and $i=k$, $\beta=0$ otherwise. At the root, summing $\phi_{R|R \leftarrow k} \cdot \bar{C}_{R,k,j \rightarrow i}$ for all k will yield the overall average values.

The equivalence of the explicit and implicit examination of all MPRs was checked during beta-testing by having several versions in which MacClade automatically calculated the values using both methods, and the values were compared and found essentially identical. For the calculations involving averages, the new mode is slightly more accurate, and so there was in some examples some minor discrepancy for those calculations. The only other differences were a result of MacClade 3's calculations overflowing because of an exceedingly large number of MPRs. This dual mode, in which both the new and old algorithms are used, is still available in MacClade 4.

ACCTRAN and DELTRAN

For characters of unordered and ordered type, ambiguities in character tracings can be resolved so as to

choose the assignments that delay or accelerate transformations (see Swofford & Maddison, 1987); these will tend to maximize parallelisms or reversals respectively. The DELTRAN option shows those most parsimonious assignments that delay changes away from the root; this maximizes parallel changes. In the example illustrated under "[Ancestral states and reconstructions](#)" on page 67, tracing number 1 is the DELTRAN tracing, as white is carried as far up the tree as possible. The ACCTRAN tracing, equivalent to tracings from Farris's (1970) algorithm, shows those assignments that accelerate changes toward the root; this procedure maximizes early gains and thus forces subsequent reversals. In the example, the ACCTRAN tracing is:



Note that this is not a single, unequivocal reconstruction, but is instead a composite of reconstructions 4–6 combined. This illustrates the fact that ACCTRAN and DELTRAN do not always choose a single one of the most parsimonious reconstructions.

ACCTRAN and DELTRAN are but two of various methods to select from among the most parsimonious reconstructions. They should probably not be used unless their treatment of parallelisms and reversals are appropriate for the biological system being considered. Before using them, heed the warnings in the section "[Which reconstructions should be examined?](#)", below.

ACCTRAN and DELTRAN reconstructions use the algorithms described by Swofford and Maddison (1987). For both, the downpass of the algorithms is done as normal. For ACCTRAN, MacClade then proceeds up the tree in a final pass that assigns to node N the states in its downpass set that are closest to the states already assigned in the final MPR set in its ancestor M. The closest states consist of the intersection of D_N and F_M if it is not empty; otherwise, they consist of D_N if the character is unordered, or the closest element in D_N to F_M if the character is ordered. For DELTRAN, MacClade does a normal nonDELTRAN reconstruction of final MPR sets, then moves back up through the tree, reassigning final sets by choosing the states in the original F_N that are closest to the revised F_M . Note that MacClade, unlike PAUP*, does not choose the lowest-valued state at the root to begin these processes. Thus MacClade's ACCTRAN and DELTRAN may not fully resolve ambiguity in the ancestral state reconstruction.

There are other options for choosing among equally parsimonious reconstructions of ancestral states. The MINF option discussed by Swofford & Maddison (1987) is not available in MacClade. Two more options are the MINSTATE and MAXSTATE options, which examine the full set of equally parsimonious states at each node, and choose the smallest and largest values, respectively. It can be shown that for ordered characters, choosing the smallest member of each reconstructed state set results in a set of assignments to the nodes that together comprise one of the most parsimonious reconstructions of ancestral states; likewise, if the largest is chosen. In MacClade, MINSTATE and MAXSTATE options are available with continuous-valued characters, and they are used in the simulation portion of the concentrated-changes character-correlation test. MacClade does not supply a direct way to obtain MINSTATE and MAXSTATE reconstructions for ordered discrete characters, but you can indirectly obtain MINSTATE and MAXSTATE

reconstructions by asking for the first reconstruction and the last reconstruction, respectively, using the **Show MPRs Mode** command in the Trace menu and then using the **Go To** command.

Which reconstructions should be examined?

When there are equally parsimonious reconstructions of the character tracing, it would be best to examine all the alternatives in full detail. However, when there are many, this may not be feasible. Which reconstructions you decide to examine will of course depend on the question you are asking. Try, if you can, to examine reconstructions that will give as different answers to your question as possible. ACCTRAN and DELTRAN yield extremes of reversals versus parallelisms. The first reconstruction versus last reconstruction in Show MPRs Mode yield extremes of lowest numbered states versus highest numbered states, with priority given to nodes closer to root. But these particular extremes may not be appropriate for your question.



EXAMPLE: Consider the character in the example file "DELTRAN ACCTRAN Example." There are 64 equally parsimonious reconstructions of character evolution for this character on the tree. If we are interested in the number of changes from wrinkled (state 2) to smooth (state 0) texture, then we might think to ignore most of the 64 reconstructions, and examine just the DELTRAN and ACCTRAN reconstructions, hoping that these would provide extreme values for the number of wrinkled → smooth changes. There is 1 DELTRAN reconstruction, which displays 0 changes from wrinkled → smooth. There are 2 ACCTRAN reconstructions, which both display 0 changes from wrinkled → smooth. Unfortunately, the DELTRAN and ACCTRAN reconstructions do not provide extreme values, for there exists one most-parsimonious reconstruction of evolution (reconstruction number 64, which can be seen by turning on **Show MPRs Mode**, and using the **Go To** command) for which there are 7 changes from wrinkled → smooth!

You should think in particular about what reconstructions would be least and most compatible with hypotheses you might be testing, and examine these. In some circumstances, the extremes of most and least compatible may coincide with the DELTRAN and ACCTRAN reconstructions, or the MINSTATE and MAXSTATE reconstructions, or they may not, depending on the biological question you are asking. Again, it is best to examine all reconstructions, if this is possible, especially if you are not certain about what reconstructions are most critically related to your biological question.

Uncertainty in the reconstruction

A reconstructed ancestral state is an estimate. Like any other estimate, its value may have uncertainty.

Equally parsimonious reconstructions

The first expression of uncertainty is ambiguity in the reconstruction; that is, the existence of multiple equally parsimonious reconstructions. Ideally, when faced with multiple reconstructions, one would examine all of them, in order to see if they differ in their implications about that aspect of character evolution of interest (D. Maddison, 1991b). Methods that examine only a subset of the reconstructions should be avoided, if possible (see further discussion in the section "Which reconstructions should be examined?", above).

Confidence

Uncertainty does not end with ambiguity in the reconstruction. Even when the reconstruction unequivocally assigns a single state to a node, this assignment may still be incorrect. Reconstructions of ancestral states are subject to error, as are all estimates of history. Studies of the reliability of ancestral state reconstructions have yielded mixed results, although it is clear that when rates of evolution are high over the time scale of the tree, error rates can be high (Holmquist, 1979; Goodman, 1981; Kimura, 1981; Tateno, 1990; Hillis et al., 1992; W. Maddison, 1995; Cunningham et al., 1998). Statistical approaches including likelihood, which have long been studied for reconstructing phylogenetic trees, have been only recently

applied to assess the behavior of parsimony ancestral state reconstructions (W. Maddison, 1995) or to serve as an alternative to parsimony (Schluter, 1995; Schluter et al., 1997; Pagel, 1999b). Whether probabilistic methods provide better estimates than parsimony remains to be seen, but they do make clear the errors to be expected. Because reconstructions of ancestral states will continue to be used in interpreting evolution ([Chapter 3](#)), it is important that their inaccuracies be appreciated.

It is to be expected that most methods will behave better as rates of evolution are lower. One's confidence in a reconstruction should depend on how many changes seem to have occurred. The fewer the changes, the more confidence one might have, especially if they are spread apart on a large tree.

However, if one's goal is to reconstruct the precise sequence of evolutionary changes in the character, *realize that the precise sequence of changes requires estimation of almost as many parameters as there are data points (observations in terminal taxa)!* It may be the case that summary statistics based on reconstructions of character changes, such as the number of steps or the ratio of gains to losses, can be more reliably used than can the precise sequence of changes in the reconstruction, because the former often requires estimation of many fewer parameters.

Even within the parsimony framework we might investigate confidence, by looking at the number of extra steps required by alternative reconstructions. In some circumstances, this gives disappointing results. For an unordered character and a dichotomous tree, placing a state at any node can give one of four results: it will be a most parsimonious assignment, or it will be less parsimonious by one, two, or three steps. The worst case (three steps) can occur only if the node is an internal node other than the root and is surrounded by nodes all agreeing on some character state. If the node is the root, then even the worst character state assignment will cost only two steps more than the best, and if the two descendants of the root disagree on their character states, the worst assignment to the root can cost at worst one step more than the best. Thus, for instance, the human mitochondrial phylogeny of Cann et al. (1987), which places Africa on one side of the root and Asia on the other, requires an equal number of steps if either Africa or Asia is placed at the root, but at worst only one extra step (out of more than 40 steps total) if any other continent is placed at the root and geographic location is treated as an unordered character (D. Maddison, 1991b).

Statistics derived from reconstructions

Several statistics concerning the number and cost of changes in a reconstruction have been proposed and are available in MacClade. For individual characters, these are the number of steps, number of changes, consistency index, retention index, and rescaled consistency index. Summed over all characters, possibly weighted by the character weights, these are the treelength, tree changes, and ensemble indices.

Steps

As noted in [Chapter 3](#), the number of steps required in a character is the sum of the costs of each of the character state transformations implied on a most parsimonious reconstruction of evolution. Except for Dollo characters and stratigraphic characters, MacClade counts the number of steps required in a character during the downpass. For unordered, ordered, and irreversible characters, a running total of steps is kept during the downpass, as indicated in the above descriptions of algorithms. For user-defined type characters, the algorithm calculates for each node the number of steps required in that node's clade by each ancestral state for the clade; once the downpass arrives at the root, the least number of steps required by any state at the root is the number of steps required in the character in the whole tree.

MacClade adds to the number of steps any steps that must have occurred within polymorphic terminal taxa (see ["Interpreting the character tracing" on page 345](#)). In general, the algorithms count the minimum number of steps that must have occurred within these terminal taxa and add this to the count of steps for the character. Thus, four steps are presumed to have occurred within a terminal taxon polymorphic for states 2, 5, and 6 in an ordered character. Similar calculations can be done for unordered, irreversible, and

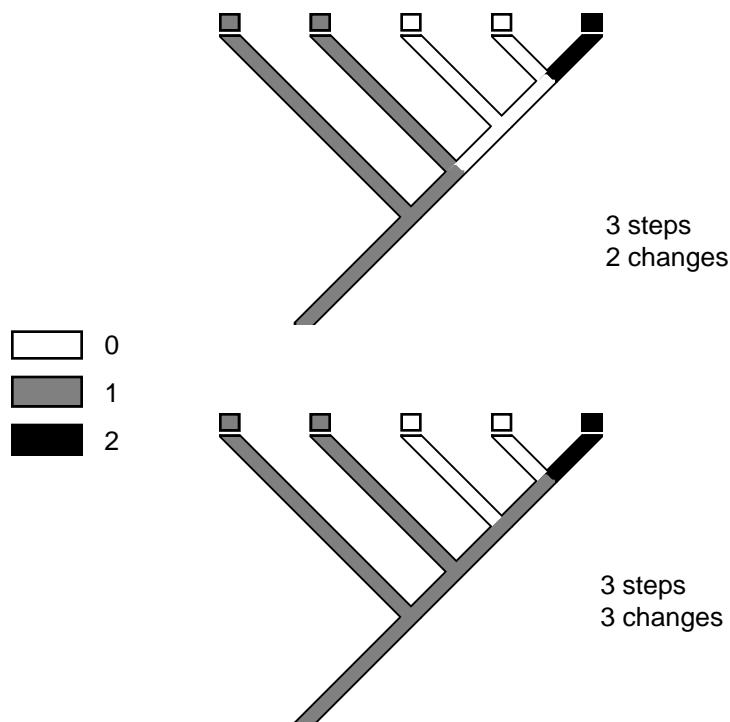
Dollo characters, but for characters of user-defined type it is very difficult to count the minimum number of steps that must have occurred within a terminal taxon if it is polymorphic for more than two states. Details are given in the description above. In these situations MacClade puts a "+" beside the count of steps and treelength to indicate that there is some uncounted length.

In [Chapter 19](#) are explained the calculations used to sum the number of steps for each character into the treelength.

Changes

In an unambiguous most-parsimonious reconstruction of character evolution, a change is required whenever a node has a different state than its immediate ancestor. The number of changes in the character is simply the number of instances in which a change is reconstructed as occurring on a branch. In MacClade, implicit multiple changes within a branch are not counted separately. Thus, if an ordered character shows a 0 to 2 change across a branch, MacClade counts only a single change. It does not suppose that state 1 is a necessary intermediate and count two changes on the branch, 0 to 1 and 1 to 2. Because of this behavior, it may be best to consider the count of changes in each character as the number of *branches* on which the character shows a change in a most parsimonious reconstruction.

If there is ambiguity in the tracing of character evolution, such that an equivocal assignment is given to some branches, then the alternative reconstructions of character evolution may have different numbers of changes. This can arise, for instance, if one reconstruction allows a single change worth two steps, whereas another equally parsimonious reconstruction allows two changes each worth one step. In the following figure, an ordered character shows three steps in each of the two reconstructions of character change, but shows two changes in the upper one, and three changes in the lower:



MacClade implicitly examines all most-parsimonious reconstructions in the character tracings using the method described in the section "[Examination of all MPRs during MacClade's calculations](#)", and determines the range of number of changes required among these different reconstructions. (If the old MPR calculation mode is used, MacClade explicitly examines all MPRs.) Therefore MacClade may sometimes

indicate a range of changes, for instance "5–7 changes". Changes within polymorphic terminal taxa are *not* counted. (Recall that their costs are counted into the number of steps.) MacClade cannot calculate the number of changes with polytomous trees or trees that have observed taxa fixed as ancestors.

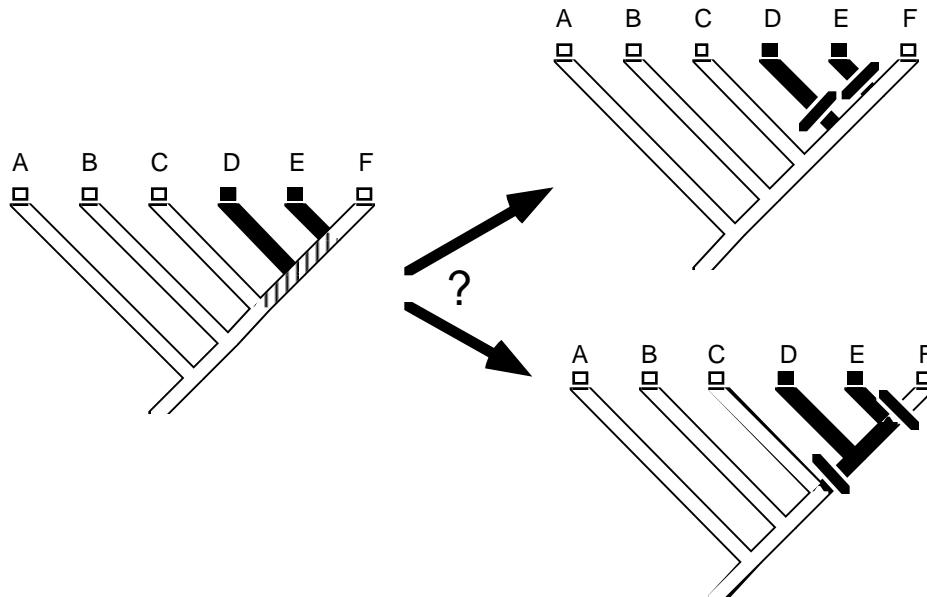
Consistency and retention indices

These indices are derived by scaling the number of steps required by a tree by the minimum and/or maximum conceivable number of steps the character could have in any possible tree. Both single-character and ensemble indices are possible. They are described further in [Chapter 19](#).

Locating and summarizing changes of character state

In several contexts MacClade examines reconstructions of character evolution and locates or counts the changes in character state that are reconstructed as occurring. For instance, this is done to put bars on the branches in Trace All Changes, and in the State Changes & Stasis chart to summarize the sorts of changes reconstructed. MacClade's calculations are described here in order to document them. See also [Chapter 19](#) for a discussion of counting the total number of changes in the tree.

Ambiguity in the reconstruction can make difficult the assessment of what changes occurred. If the MPR set at a node has multiple states (an equivocal assignment), then whether a change is placed on the branch between this node and its ancestor may depend upon which of the alternative states is assigned to the node. For instance, in the following example, whether a change is reconstructed on the branch below the clade D-E-F depends on which of the two equally parsimonious reconstructions at right is chosen:



The two reconstructions also disagree on the nature of the changes: The top reconstruction suggests two white to black changes; the bottom suggest a white to black and a black to white change.

In Trace All Changes

In MacClade's Trace All Changes facility, the amount or identities of changes on each of the branches of the tree are marked. To deal with ambiguity in the reconstructions, four different options are offered by MacClade that vary in how they assess whether changes are reconstructed as occurring on particular branches. These are described further in [Chapter 18](#). Note that changes on branches participating in a soft polytomy

are never recorded because of difficulties in interpreting these changes. The four different options and their calculations are:

1. Show unambiguous changes only. An unambiguous change is counted as occurring on a branch if the sets of states reconstructed at a node and its ancestor do not overlap at all. This option will show no changes in the example above. If weighted then the distance between these sets is used as the amount of change.
2. Show almost all possible changes (most ambiguous and unambiguous). A possible change is counted if the MPR state sets of a node and its ancestor are not identical. This option will show all four possible changes in the example just above. (The weighted amount of change cannot be calculated, because with this criterion it is not clear what changes are occurring and thus their weights cannot be considered.)
3. Show the maximum amount of change allowed by parsimony. This choice finds all instances in which parsimony allows a change on the branch. For node N and unordered and ordered characters, MacClade compares N's uppass set and downpass set. If these sets intersect, then there are no changes allowed on N's branch; if they do not intersect, then the distance between the sets is the maximum amount of change allowed on the branch. For other character types MacClade uses the implicit examination of all most-parsimonious reconstructions (["Examination of all MPRs during MacClade's calculations"](#), above) to see if at least one of these reconstructions places a change on the branch. (If the old MPR calculation mode is used, MacClade explicitly examines all MPRs.)
4. Show the minimum-average-maximum amount of change. The minimum and maximum amounts of change in the character on the branch are determined by implicit examination of all most-parsimonious reconstructions (["Examination of all MPRs during MacClade's calculations"](#), above). (If the old MPR calculation mode is used, MacClade explicitly examines all MPRs.) The average is calculated in one of two ways: average over all reconstructions, and average over all classes of changes. In the first option, all most-parsimonious reconstructions are implicitly or explicitly examined. The total number or amount of change on a branch is summed over all such reconstructions, and divided by the number of such reconstructions. Thus, in the example of six reconstructions beginning on [page 68](#), the branch leading to taxon 3 shows changes in two out of the six reconstructions (numbers 5 and 6), and thus the average amount of change is 1/3 change per reconstruction. In the second option, all distinct classes of change reconstructed on the branch are examined. In the same example, there are three classes of change, either no change (reconstructions 1–4), change from gray to white (reconstruction 5) or change from black to white (reconstruction 6). Two of these classes show a change, and thus the average amount of change is 2/3 changes per class.

In State Changes & Stasis chart

The State Changes & Stasis chart counts the number of instances in which a reconstructed change is from state 0 to state 1, the number from state 1 to state 2, and so on. It therefore counts the number of each sort of character state transformation reconstructed on the tree.

The following options are available to count changes for the State Changes & Stasis chart. As with Trace All Changes, changes on branches participating in a soft polytomy are never counted.

1. Count unambiguous changes only. An unambiguous change from state s to state t is counted as occurring on a branch N if the single state s is reconstructed at N's ancestor and the single state t is reconstructed at N. If the MPR set at either node is equivocal, no change is counted.

An unambiguous change from s to t is counted as occurring *within* a polymorphic terminal taxon if the terminal node is assigned a single state s but the terminal taxon had also the single other state

- t. If the terminal node is assigned a single state and the character is ordered, irreversible, stratigraphic, or Dollo, then a series of unambiguous changes are counted if the terminal taxon is polymorphic for more than two states. In this situation it is presumed that changes occurred in an ordered fashion from the assigned state at the terminal node to each of the other states observed in the terminal taxon.
2. Count minimum and maximum changes. MacClade performs implicit or explicit examination of all most-parsimonious reconstructions to see in each what changes are reconstructed. In each reconstruction the branches are examined to see what changes are placed on the branch.

Within polymorphic terminal taxa changes are counted as follows. For unordered characters, changes from the resolved state at the terminal node to each of the observed states in the terminal taxon are counted. Note that this is only one of the possible sets of changes that might have occurred. *Because MacClade does not examine all possible pathways of change within each terminal taxon, it cannot give the exact minimum and maximum number of changes of various sorts*, and therefore gives a warning whenever the taxon has more than one state beyond that assigned to the terminal node. For ordered, irreversible, stratigraphic and Dollo characters, changes within polymorphic terminal taxa are counted as above for unambiguous changes.

After each most-parsimonious reconstruction is examined, MacClade sees whether the number of each type of transformation (e.g., state 0 to state 1) is larger than the maximum found among previous reconstructions, or smaller than the previous minimum. If so, then the maximum or minimum is reset. Thus, MacClade finds among all reconstructions the minimum and maximum numbers of each type of transformation reconstructed.

3. Count average number of changes. MacClade proceeds as for the minimum/maximum algorithm, except that it does not store the minima and maxima while examining the reconstructions, but rather calculates the average, over all the equally parsimonious reconstructions, of the number of each type of transformation.

Polytomies and their difficulties

Soft versus hard

Trees that have polytymous nodes, that is, nodes with more than two descendent nodes, are particularly troublesome in phylogenetic analyses. The polytomies may be interpreted in two different ways (W. Maddison, 1989): (1) as regions of ambiguous resolution ("soft" polytomies), or (2) as multiple speciation events ("hard" polytomies). Although in most cladistic literature polytomies are interpreted in the first sense, most algorithms for reconstructing character evolution that have been presented in the literature and, as far as we know, in other computer programs use the second interpretation, because any daughter nodes of the polytomy that share a state different from that of the polytomy are presumed to have acquired it independently (W. Maddison, 1989). In contrast, soft polytomies represent uncertain resolution, in that the polytomy is taken as indicating a set of alternative dichotomous resolutions. Soft and hard polytomies differ in the interpretation of their display in MacClade, as discussed in "Interpretation of character tracings", in [Chapter 18](#).

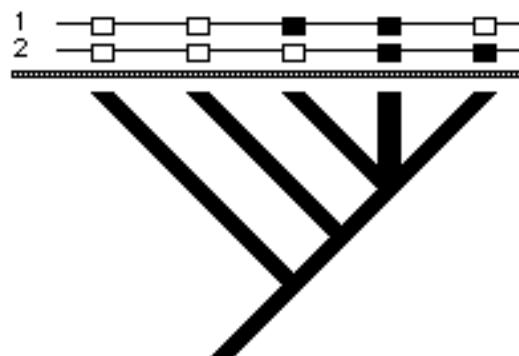
Reconstructing character evolution

The description of algorithms above indicated how MacClade reconstructs character evolution on polytomies. For hard polytomies, the polytomy is fixed and MacClade considers each descendent node as independent in assessing what states would be parsimonious. For soft polytomies, MacClade reconstructs character evolution as if the polytomy were allowed to be resolved most favorably for that character.

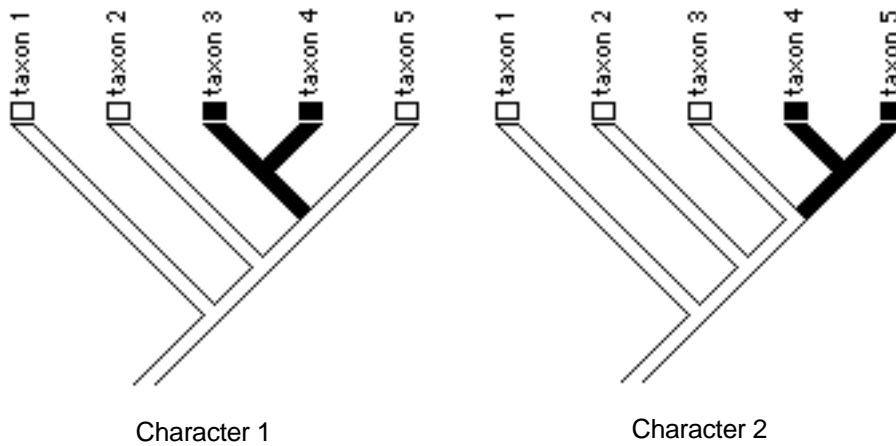
It is particularly difficult to reconstruct whether a character changed state on the branches arising out of a soft polytomy. For this reason, MacClade indicates no changes on any such branch in its Trace All Changes feature ([Chapter 18](#)). The problem arises in that, under different dichotomous resolutions of the polytomy, different changes may be assigned to these branches. Perhaps the best solution would therefore be to resolve the polytomy into all its dichotomous resolutions, reconstruct the changes occurring in each case, and summarize the alternative possibilities by showing the range (over the various resolutions) of the amounts of change placed on these branches. Note, however, that some of the resolutions may be highly unparsimonious with respect to the evolution of the character concerned. As well, there may be variation among resolutions in the changes reconstructed on branches other than those directly involved in the polytomy. Recall that MacClade's character tracings reconstruct evolution on soft polytomies assuming the most favorable resolutions for the characters considered. In the future MacClade may incorporate features to examine alternative dichotomous resolutions and summarize the varying branch lengths, but currently you will have to do this by hand, perhaps by asking MacClade to generate a set of random dichotomous resolutions of the tree ([Chapter 23](#)).

Treelength

When MacClade reconstructs character evolution as if a soft polytomy were allowed to be resolved most favorably for that character, it allows different characters to implicitly resolve the polytomy in different ways. This leads to difficulty in interpreting treelengths (W. Maddison, 1989). The problem can be seen by an example. Suppose this were our tree with two characters as shown:

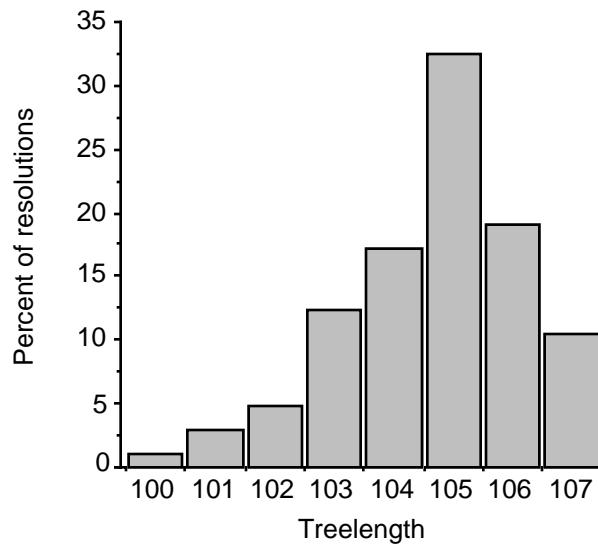


MacClade reconstructs each character's evolution allowing the polytomy to be resolved most favorably for each. Thus, for each of the two characters a different resolution is chosen and one change will be counted as follows:



Summing the number of steps counted for each character will give an apparent treelength of 2, but in fact there does not exist a dichotomous resolution with fewer than three total steps. In general the treelength obtained by summing character steps can be misleadingly low when there are soft polytomies. (For this reason MacClade puts a "+" after the treelength whenever there are soft polytomies, just to warn you that the treelength may be misleadingly low.)

There are two possible meanings of the length of a tree with soft polytomies. One could argue, as did W. Maddison (1989), that the true length of a tree with soft polytomies, that is, the number of steps required by the polytymous tree, should be considered the length of the most parsimonious dichotomous tree consistent with the polytymous tree. This would be the "minimum number of steps required by the polytymous tree". To find such a most parsimonious dichotomous tree, a tree search may be required. However, considering a tree with soft polytomies to be simply a summary diagram of a larger number of implied dichotomous trees, one could argue that the treelength of this collection is not the length of the shortest tree contained therein, but rather a set of treelengths. For example, consider the data file called "Polytomy Example" distributed with MacClade. In the single tree stored in that file, there is a polytomy with five elements. This polytymous tree thus represents 105 dichotomous resolutions. The treelength presuming soft polytomies is given by MacClade as 91+. In fact, the histogram of length of all included dichotomous trees is:



Of the 105 trees, there exists one of length 100, some of length 107, and quite a few (34) of length 105. If one were to ask, "What is the length of that polytymous tree?", one could answer either "100" or "lengths range from 100 to 107, as shown in the histogram above". See [page 395](#) for further consideration of this example.

Zero-length branches versus consensus trees

In programs like PAUP* that reconstruct phylogenetic trees, polytomies can arise either because branches with no support ("zero-length") are collapsed in the course of the search process, or because several different trees are combined into a consensus tree. In both cases, the polytomies should be considered soft, but the two sorts of trees should be treated differently. The reason is that all possible dichotomous resolutions of the polytomy arising by zero-length collapsing should be equally parsimonious, but this is not necessarily the case for consensus trees. Even though the consensus may be compatible with the parsimonious trees that went into it, there may have been other dichotomous trees that would be compatible with the consensus but were not among the most parsimonious. For example, consider the example file "Hamamelidae", containing data on plant structure (from Donoghue and Doyle, 1989). The strict consensus tree of the most parsimonious trees is compatible with about 41,910,000,000,000 dichotomous trees (calculated using

the formulae provided by Felsenstein, 1978a), of which only 54 dichotomous trees are of shortest length. After trees are condensed to a consensus, the information as to which dichotomous trees went into it is lost. MacClade might choose as a most favorable resolution (in reconstructing character evolution on a soft polytomous consensus tree) one that was not among the most parsimonious, and you may get a reconstruction of character evolution that would not have been allowed by any of the dichotomous trees that made the consensus. Because consensus trees have lost the information about what trees went into them, reconstructing character evolution and counting treelength on them should be avoided, or at least done with much caution.

Looking at alternative resolutions

Because of the problems with soft polytomies, it is generally advisable to examine alternative dichotomous resolutions of the polytomies. (It is *not* advisable to switch to the hard polytomies interpretation, if the polytomies truly represent uncertainty.) A common situation will be similar to the following. You want to examine the evolution of some particular character, say chromosome fusions, on a phylogeny, inferred using PAUP*, based on morphological data. But suppose PAUP* (or another tree-searching program) does not generate a single dichotomous tree. Instead, suppose it generates a tree with many polytomies due to collapsing of zero-length branches. If you examine the character's evolution on this polytomous tree, your results could be biased by showing too few changes in the character (if you use the soft polytomy option) or too many changes (if you use the hard polytomy option).

There are two ways to solve this problem. MacClade can generate random dichotomous resolutions of the current tree on screen. You might generate a number of such trees and examine the character's evolution on each. Or, you might ask for a chart summarizing the statistic of interest over trees consisting of random dichotomous resolutions of the single polytomous tree on the screen. In doing this, you would be sampling from the set of all possible resolutions, in order to see how the statistic of interest might vary over these resolutions. You might see, for instance, that under some dichotomous resolutions, as few as four steps are required in the character, whereas under other resolutions, as many as eight steps are required.

Another way to solve the problem is to ask a tree-searching program like PAUP* not to collapse zero-length branches. However, this could slow the tree search and use up much more memory. It would be particularly damaging to the efficiency of the tree search if there were multiple equally parsimonious trees, each of which was polytomous (with zero-length branches), as is most often the case.

If a tree-searching program generates multiple equally parsimonious trees, each of which is polytomous, what are you to do? It would be very laborious to place each of one thousand possible polytomous trees on screen, and ask for a chart summarizing the statistic of interest over one thousand random dichotomous resolutions of each tree. Instead, you can examine one randomly chosen dichotomous resolution of each of the one thousand possible polytomous trees. (This can be done in MacClade using the Resolve Polytomies button that is found in some of the charting dialog boxes.)

Although examining random dichotomous resolutions of polytomies might be appropriate if the polytomies arose by the collapsing of zero-length branches, it is a dangerous practice if the polytomies arose in the formation of a consensus tree, as described above. It is best to examine directly the trees that went into forming the consensus tree. MacClade's facilities for examining multiple trees and presenting summaries in chart form can be used for these purposes.

Uncertainty in the phylogenetic tree

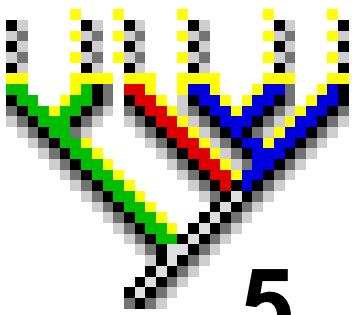
Rare will be the case that we know a phylogenetic tree with certainty. If there is uncertainty in the phylogenetic tree, how will this affect our reconstructions of character evolution? We have already described reconstructing evolution on trees with soft polytomies, and we advised examining alternative dichotomous trees resolved from the polytomy. If there are multiple equally (or almost equally) parsimonious

trees, then reconstructing character evolution on each allows one to see whether the different trees yield different reconstructions of character evolution (D. Maddison, 1991a,b; Swofford and Maddison, 1992). MacClade has features specifically designed to facilitate dealing with multiple trees, especially the charting functions.

Maddison et al. (1984) presented a number of simple rules that indicate whether and how much certain changes to a tree might affect a parsimony reconstruction of ancestral states. Although these rules were presented in the context of outgroup analysis, they can be applied in general to reconstructions of ancestral states for unordered and ordered characters:

1. If two terminal taxa that branch from a lineage one after the other have the same state, then changes in the tree above these have no effect on states reconstructed below them, and vice versa.
2. If the entire tree is rerooted, then the states at a node (other than the root) are unaffected.
3. If a single clade is removed from or added to any part of the tree, then the state set at a node before and after the move must overlap, as long as the node itself was not removed.
4. If a single clade is picked up from any part of the tree and dropped somewhere else, then the state set at a node before and after the move must overlap, unless the node was an immediate ancestor of the clade. (In this instance, picking up the clade would destroy the node.) This rule holds for binary characters, but we have been unable to prove it for multistate characters.

These rules provide guidelines for only some simple examples of uncertainty in tree structure; other cases of uncertainty are not amenable to simple rules and will require examination of multiple trees.



GETTING STARTED WITH MACCLADE

5

MacClade shares with most other Macintosh programs certain conventions – the appearance of windows, drop-down menus, dialog boxes, scroll bars, and so on. We will assume that you have basic familiarity with the Macintosh and so we will not detail all the rules for using menus and editing text, for example. This chapter discusses basic issues of hardware and software requirements of MacClade, how to start MacClade, messages MacClade may display, and how to set default values for MacClade options.

Hardware and system software specifications

MacClade 4 works on any Macintosh (including PowerMacs, PowerMac G3, PowerMac G4, iMacs, Powerbooks, iBooks, Quadras, Centris, and Performas, among others) that can run System 7.5 or later. A hard disk and CD drive are required. The amount of memory you will need in your computer depends upon the nature of your data and tree files; see the section "[Computer memory and MacClade's limits](#)", in this chapter, for details. In general, MacClade requires 4Mb or more of RAM devoted to itself. MacClade will work with any Macintosh-compatible printer that can print graphics (e.g., dot matrix, ink jet, laser); however, some of its tree-printing features require a PostScript® printer.

MacClade 4 requires System 7.5 or later, and is compatible with Systems 8 and 9. Although MacClade 4 should run under MacOS X, it will not take advantage of all of MacOS X's features. We hope to improve this after MacOS X's release.

MacClade 4 can also run on a computer using the Microsoft Windows® operating system within a MacOS emulator, as discussed below.

Installing MacClade

MacClade and associated files are distributed in folders contained on this CD.

The CD contains two versions of MacClade. The copy whose name ends in "PPC" is the PowerPC version of MacClade, and is the version that should be used on all modern Macintoshes, including all PowerMacs and any other Macintosh with a PowerPC processor (including G3 and G4 processors). The copy whose name ends in "68K" is the version for older Macintosh models using 68020, 68030, or 68040 processors, such as Quadra and Centris, as well as for some MacOS emulators like Executor (see "[Using MacClade under Microsoft Windows®](#)").

The folder "MacClade Manual" contains documentation, including this manual, as well as a folder of examples. The Examples folder contains files used as some of the examples in this manual.

To install MacClade, simply drag the appropriate copy of MacClade (PowerPC or 68K) from the CD to your hard disk.

WARNING: The method used by MacClade to personalize your copy may conflict with some system extensions. We currently know of one conflict: you will not be able to successfully install and start MacClade if you have the CleverContent extensions enabled on your computer. If you have difficulty starting MacClade, then check to see if any CleverContent files are present in the Extensions folder of your System Folder. If they are present, then either disable

them using the Extensions Manager control panel or remove them from the Extensions folder, restart your Macintosh, and then reinstall and restart MacClade. Once you have successfully started MacClade, you should be able to reinstall CleverContent and then use MacClade.

There are also classroom versions of MacClade in the MacClade Classroom Version folder. Install these on a computer by simply dragging the appropriate copy (PowerPC or 68K) onto the computer's hard disk.

Using MacClade under Microsoft Windows®

MacClade is a MacOS program, designed to run on an Apple Macintosh computer. However, you can use it on other computers if they have MacOS emulators. Here we will discuss emulators on the Microsoft Windows operating system; there may be emulators for other operating systems, but we have no experience with these.

There are several MacOS emulators available for computers running Windows. They include Executor (<http://www.ardi.com/>), SoftMac 2000 (<http://www.emulators.com/>), and the vMac Project (<http://www.leb.net/vmac/>). Of these we only have personal experience with Executor.

Using Executor to run MacClade under Windows

ARDI's Executor (<http://www.ardi.com/>) will allow you to run the 68K version of MacClade. We do not have extensive experience ourselves using MacClade in this context, but we know of several MacClade 4 users who successfully use the program in Executor. MacClade does not run perfectly under Executor, and there are limitations and anomalies you may encounter, although it does work remarkably well. For the latest information about using MacClade under Executor, see the MacClade web site (<http://phylogeny.arizona.edu/macclade/macclade.html>).

To install MacClade into the Executor environment, insert the MacClade CD into the computer. In the Program folder will be two copies of MacClade: mc68k.hqx is a bin-hexed archive that contains the 68K version of MacClade; mcppc.hqx contains the PowerPC version. They also contain the example files. Copy the file mc68k.hqx file to the App directory in the directory containing Executor. Start up Executor; within Executor, start up Stuffit Expander. In Stuffit Expander, open the file mc68.hqx. This will unstuff the archive, and you will be left with a directory containing MacClade and the examples in your App directory. You will then be ready to start MacClade from within Executor.

Updates to MacClade

If there are updated versions of MacClade, information will be posted on MacClade's World Wide Web site (<http://phylogeny.arizona.edu/macclade/macclade.html>). At times there may be files posted which you can download to update your copy of MacClade.

Full and Classroom versions of MacClade

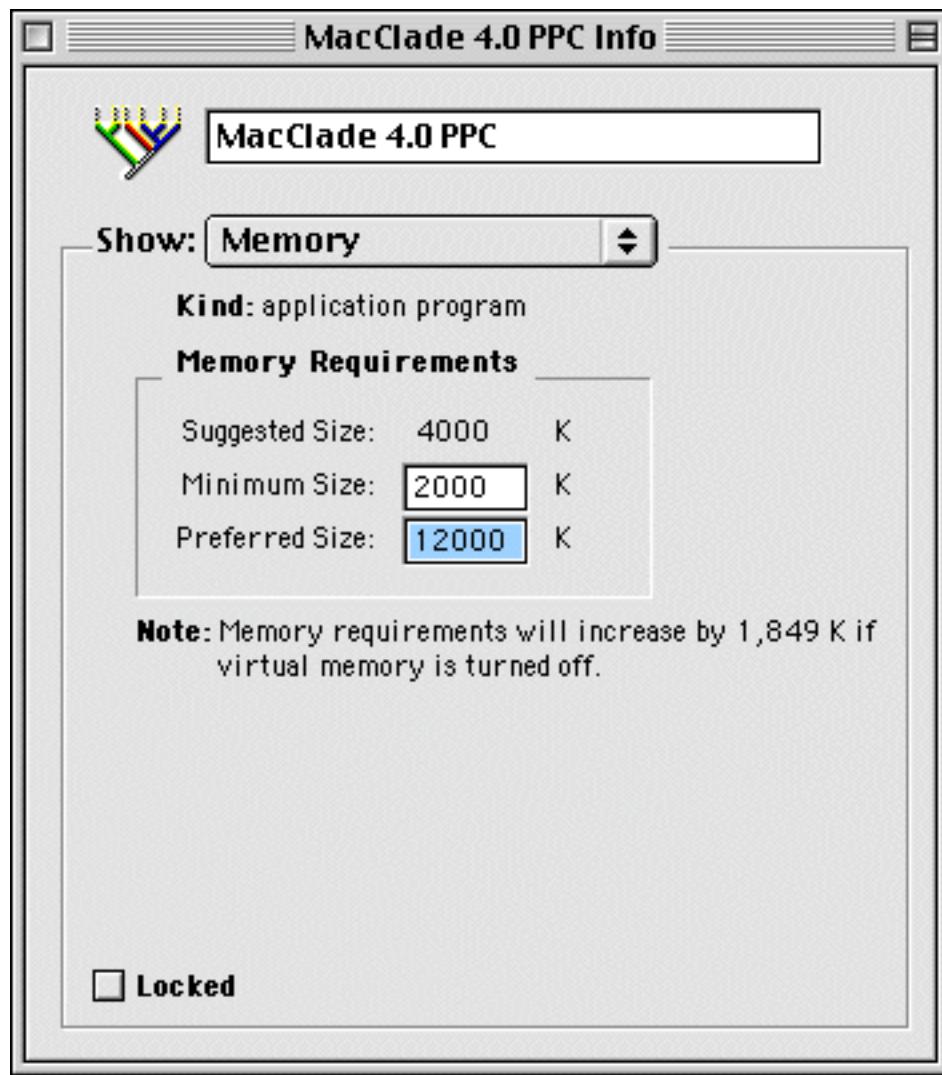
The full and classroom versions of MacClade are almost identical. They differ in only two aspects: (1) the color of the opening screen and the **About MacClade** dialog box, (2) access to the **Editing Restrictions** dialog box in the full version, with the classroom version not having access and being subject to the settings in this dialog box. Editing restrictions are described under "[Editing restrictions in the classroom version" on page 123](#).

Computer memory and MacClade's limits

MacClade 4 can process data matrices with up to 1500 taxa and 16000 characters, if there is enough computer memory available. It is not easy to give exact limits on how many taxa and characters can be accommodated within a certain amount of computer memory, as it depends upon the number of stored trees, presence of pictures, whether or not you ask MacClade to calculate a chart, and other factors. Some operations, such as use of the pairwise alignment tool in the editor, require much more memory than others. Fewer characters could be accommodated if data have more than 10 states per character (protein or extended standard formats).

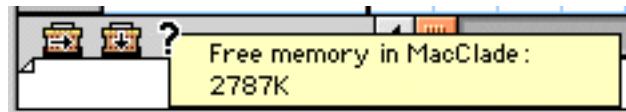
You have some control over how much memory is allocated to MacClade. Stored in each program is a number indicating how much memory the program will request from the computer for operation. MacClade asks for 4Mb, an amount that will allow MacClade to work on reasonably large data files. For example, with this amount of memory MacClade can easily handle 200 taxa and 500 characters in a morphological matrix, or 50 taxa and 2000 characters of DNA sequence. However, MacClade requires more memory for larger data sets, and for some specific tasks (in particular, some of the sequence alignment tools; see [page 251](#)). Therefore, if you find that MacClade does not have enough memory, try changing MacClade's request by selecting MacClade (make sure it is not running!) in the Finder, choosing **Get**

Info under the **File** menu, and editing the Preferred Size under the Memory panel, as shown in the following example:



The next time you start up MacClade, the MacOS will reserve this larger amount of memory for MacClade.

NOTE: In the tree window, to see amount of available memory, click on message box at the lower left of the window. In the editor, click on the empty space to the right of the buttons to see amount of available memory, as shown below:



Starting MacClade

MacClade can be started in several different ways. If you double-click on MacClade in the Finder, MacClade will begin and present you with the **Open File** dialog box, which allows you either to create a new

file (with the New button) or open an existing file (with the Open button). MacClade can also be opened by double-clicking on either a MacClade data file or stationery pad:



If you double-click on the icon of a MacClade data file in the Finder, this will start MacClade and open the selected data file. If you select more than one file in the Finder, MacClade will open only one of them. If you select a file in the Finder and give the **Print** command, MacClade will start, open that file, but not print (as MacClade doesn't know what aspect of that file you wish to print).

Data files are any files that contain specifications of the characteristics of taxa; they may also contain trees. In contrast, tree files contain *only* trees. Note that the icons for MacClade's tree files are exactly the same as MacClade's data files. If you ask MacClade to open a tree file, without first having asked it to open a data file, MacClade will give a warning that this cannot be done.

NOTE: *If the icons for your MacClade preferences or graphics files look like blank pages, then this means that the disk has had on it previously an older version of MacClade. In order for the new icons to be displayed, you may need to rebuild the Desktop file for that disk. This can be done by holding down the Option and Command (⌘) keys while the disk is first being read in or as the Finder is first opening; see your Macintosh manual for details.*

If you double-click on a **stationery pad** of a data file, MacClade will treat the file exactly as a normal data or tree file, except that it will open the document as a new untitled and unsaved document, leaving the stationery pad untouched on the disk.

MacClade's behavior on opening any other type of file varies. If the file opened is a MacClade **preferences file**, MacClade will start and use the settings stored in that file as the defaults.

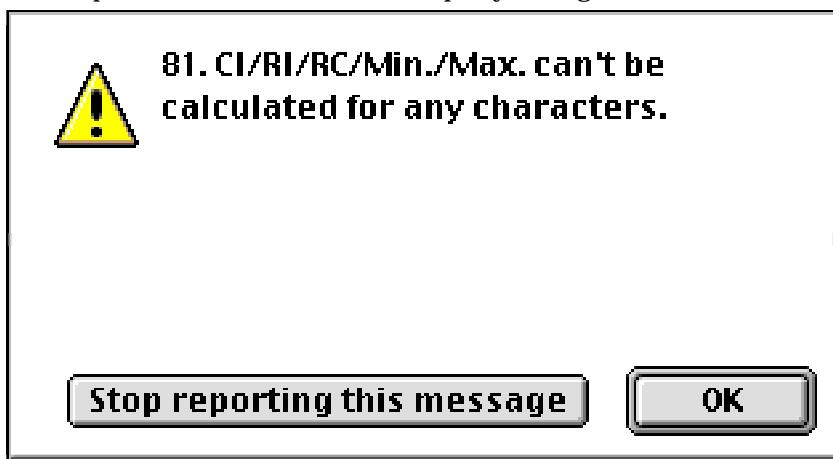
To open a data file created by another program such as PAUP*, see the instructions in [Chapter 6](#) and [Chapter 8](#).

Getting help

There is no built-in help facility in MacClade. Use this manual for information about using MacClade; you might also want to check MacClade's World Wide Web site (<http://phylogeny.arizona.edu/macclade/macclade.html>) for any updates to the documentation.

Messages

In the course of using MacClade, you may receive various warnings or messages about incorrectly formatted files, or illegal assumptions, or whatever. For example, you might encounter this:



If you choose OK, then MacClade will present you with the same warning again, if the appropriate situation arises again. However, if you choose Stop reporting this message, then MacClade will not redisplay that particular message again. This is particularly convenient if you are being warned many times about a particular problem. If you do disable a warning message in this way, there are two ways to tell MacClade to redisplay the message when appropriate: either quit from MacClade and restart the program, or hold down the Option key and choose **Restore Messages** from the **File** menu. All disabled messages will thereby be enabled.

For some of its statistics, such as treelength, MacClade may put an asterisk or other symbol beside the number, indicating that there may be problems underlying the statistic (for instance, that treelength is difficult to interpret with soft polytomies). The infinity symbol (∞) may appear if a number is undefined or negative. When a number is too large to represent with the current options, asterisks (****) may appear. By touching on the number, a message may appear that describes the problem. Problems are discussed in greater detail in the relevant sections in this manual.

Setting preferences for MacClade's options

MacClade has hundreds of options that may or may not be in effect at any one time. You can choose whether polytomies are to be interpreted as hard or soft, whether treelength calculation is turned on, whether the grid is shown in the data editor, which colors are used in the tracing of character evolution, and so on. You can change these options using various dialog boxes and menu items in MacClade.

How MacClade remembers options

When you begin a new file, MacClade sets the options to the default values. These default values are either the factory defaults, or your own particular preferences. For example, when you start a new file, the data editor will be displayed using the font Geneva, unless you have set your preference to Times, in which case the data editor will be displayed in Times. MacClade stores your preferences in a special file in the Preferences Folder within the System Folder called MacClade Preferences, and refers to this file when the program starts. The means by which you set preferences are detailed in the next section.

When you open an old file, MacClade will, in general, use the values of the options stored in the data file, overriding your preferences. For example, if you had saved a file in MacClade 4 with Helvetica as the data editor font, then that file will display the data editor in Helvetica, even if the font specified by your prefer-

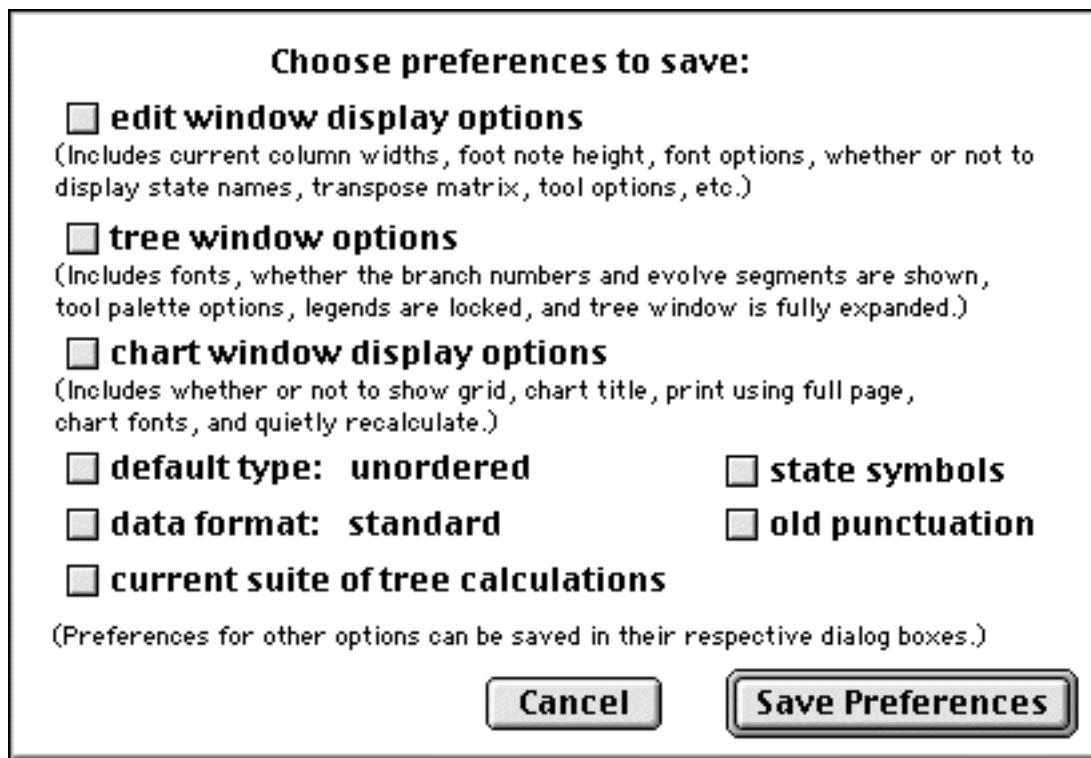
ences is Times. However, MacClade will not always use the values of options stored in data files, for two reasons. First, if the old file was not saved by MacClade 4, then the values of the options may not be stored in the file. The values of most options are included in the data file only when the file is saved by a current version of MacClade. Second, there is a small set of options whose values are not stored in each file. (The values for these options can be stored only in the program's preferences file.) For these options, the values in use will not change when you open an old file, even if that file was saved when other values were in effect. Designating preferences is the only way for you to tell MacClade to remember your choices in these exceptional groups of options, as the values for these options can be saved only in a preferences file, not in a data file.

If you start up MacClade by double-clicking on a preferences file in the Finder, MacClade will use that preferences file for defaults, *no matter what the file's name is*. Thus you can store several alternative preferences files, one for your molecular data, one for your flower morphology data, and so on. To save alternative preferences files, choose the **Save Preferences** menu item from the **File** menu, or press on the Save Preference buttons in the appropriate dialog boxes, then go into the System Folder and rename this preferences file to something other than "MacClade Preferences" (e.g., "MacClade DNA Preferences"). This protects the file from being overwritten the next time you save preferences, and your MacClade DNA Preferences can be accessed as described by double-clicking on its icon. (You might also use stationery pads for data files, as described in [Chapter 6](#). These store most, but not all, preferences.)

Designating preferences

You can tell MacClade what you want its default values to be by using the Save Preferences buttons and menu items. Many dialog boxes in MacClade have a Save Preferences button. Pressing one of these will designate the settings currently chosen in the dialog box to be MacClade's default values. (If you do not designate your own preferences, MacClade will use its own factory defaults.) The preferences will be saved in a file called "MacClade Preferences" in the System Folder.

In addition, preferences for several other options, including some aspects of the editor, chart window, tree window, the default type, and the data format, can be set using the **Save Preferences** dialog box of the **File** menu:



To use this dialog box, check off those items you want saved, and press the Save Preferences button; the settings for those items checked will then be saved.

NOTE: There are two levels of defaults for character types (see [Chapter 15](#)). Each data file has its own default type, such that any new characters created will be assigned that type. This file default is set in the **Type Edit** dialog box in the **Characters** menu. There is also a preference for the default type. The factory setting for this program preference is "unordered". Thus, if you create a new file, the default type of any character will be unordered. However, you can change this program preference by first changing the file default in the **Type Edit** dialog box, and then using the **Save Preferences** item of the **File** menu to save the preference. (You cannot do this if the file's default type is user-defined or stratigraphic.)

NOTE: To save all current options as defaults, hold down the Option key and choose **Save All Preferences** from the **File** menu.

Using MacClade with other programs

As you use MacClade on your computer you will find a need to share information between MacClade and other computer programs. Word processors and spreadsheet programs may be used in editing the data file, other phylogenetics programs like PAUP* may be involved in analyzing the data, and graphics programs may be used to modify graphics files made by MacClade. Various sections of this manual describe how MacClade can share information via files.

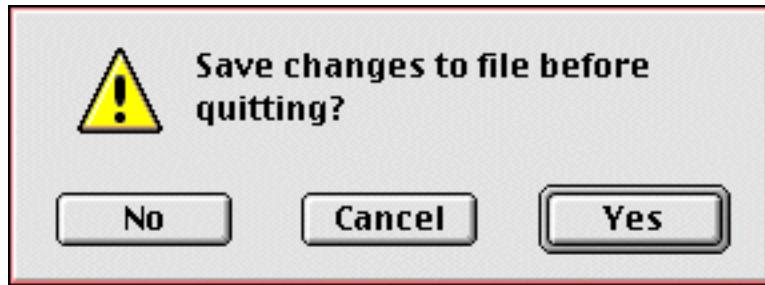
We envisage MacClade being used closely in connection with PAUP*, a program for phylogenetic analysis developed by David Swofford (1991, 2000). [Chapter 26](#) discusses issues concerning the use of MacClade

and PAUP* together. Of special concern are slight differences in calculations and file format, and the caution needed in using PAUP* and MacClade simultaneously.

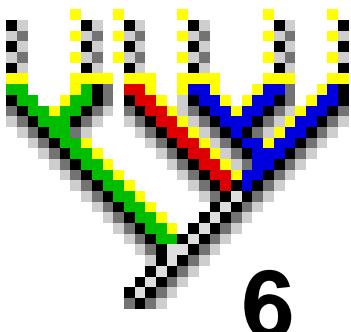
[Chapter 8](#) discusses importing and exporting data files from other programs including other phylogenetics programs and sequence alignment programs. [Chapter 25](#) discusses MacClade's saving of results in text and graphics files that can be edited by word processors and graphics programs.

Quitting from MacClade

To quit from MacClade, choose **Quit** from the **File** menu. If you have made changes (such as editing data, rearranging the tree, changing assumptions, and so on) since the last time you saved, MacClade will ask if you want to save the changes before quitting:



If you press the Yes button, then MacClade will save changes and quit; if you press No, MacClade will not save the changes and quit; if you press Cancel, MacClade will not save the changes and not quit.



MANAGING DATA FILES

In this chapter we review some basic aspects of managing files.

Data files are files that contain a data matrix; they may also contain trees. Tree files contain only trees. For some additional information about tree files in particular, see [Chapter 17](#).

Creating a new MacClade data file

To create a new MacClade data file, MacClade must be running and no data file can be open. There are two ways to ask MacClade to create a new data file. First, you can use the New button in the **Open File** dialog box. This dialog is presented to you when you start MacClade without specifying a data file, or when you choose **Open File** from the **File** menu in MacClade. Thus you can create a new file by double-clicking on the MacClade icon in the Finder, then choosing the New button in the **Open File** dialog box. Second, there is also a **New File** menu item in the **File** menu of MacClade. If the **New File** or **Open File** menu items are grayed, then a data file must still be open; you must close it before attempting to create or open another file. Whether you use the New button or menu item, you will be presented with a blank data editor with one taxon and one character.

To choose the kind of data (standard, extended standard, DNA, RNA, protein) included in the data matrix, select the appropriate item from the **Data Format** submenu under **Characters** (see the section "[Data formats](#)" of [Chapter 12](#) for more details).

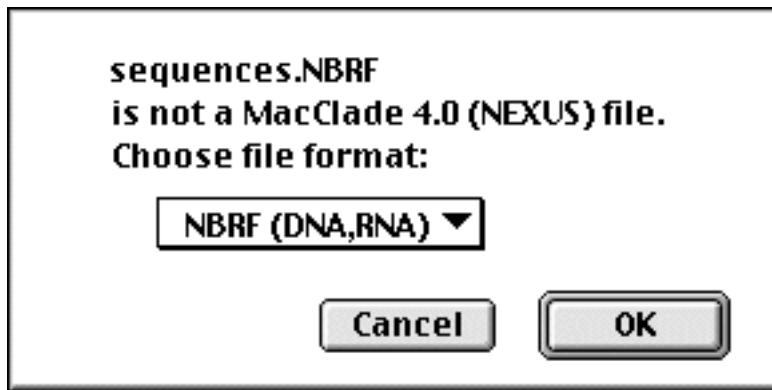
Opening an existing data file

How you open an existing data file depends on whether the data file is a MacClade data file (one with a MacClade file icon), or a data file originating from some other source (such as a word processor or PAUP*). If the data file is a MacClade data file (that is, it has a MacClade file icon), it can be opened directly from the Finder by double-clicking on it. The Macintosh will know to start MacClade and will do so automatically. If the data file is not a MacClade data file, or even if it is, you can open it by first starting MacClade, then using the **Open File** item in the **File** menu, then choosing the file in the dialog box. You can also drop the file onto the MacClade application in the Finder to open the file.

As the file is being read in, you will see a box with a status bar on it showing the progress of the reading. You will also note that the cursor changes as the file is read; each type of cursor indicates which part of the file is being read; see [Chapter 7](#) for details.

If you decide to abort the reading of a file, click on the Stop button or hold down the Command (**⌘**) key and type a period (.)

If MacClade detects that the file being read is not a standard MacClade file (perhaps it is a HENNIG86 or PHYLIP file), MacClade will present you with a dialog box asking you what type of file it is. Select the appropriate file format from the pop-up menu:



MacClade can open and understand text files in a number of formats, including those of MacClade 1.0, 2.1, HENNIG86/NONA, PHYLIP, NBRF, and MSF/GCG. (MacClade 4 uses the same file format as MacClade 3, with a few enhancements, and so it should be able to read NEXUS files saved by MacClade 3.) Also, some sorts of simple tables can be understood by MacClade. For full details on the files MacClade can import, see [Chapter 8](#).

On opening a file, MacClade will go automatically to either the tree window or the data editor window, whichever was in use when MacClade last saved the file.

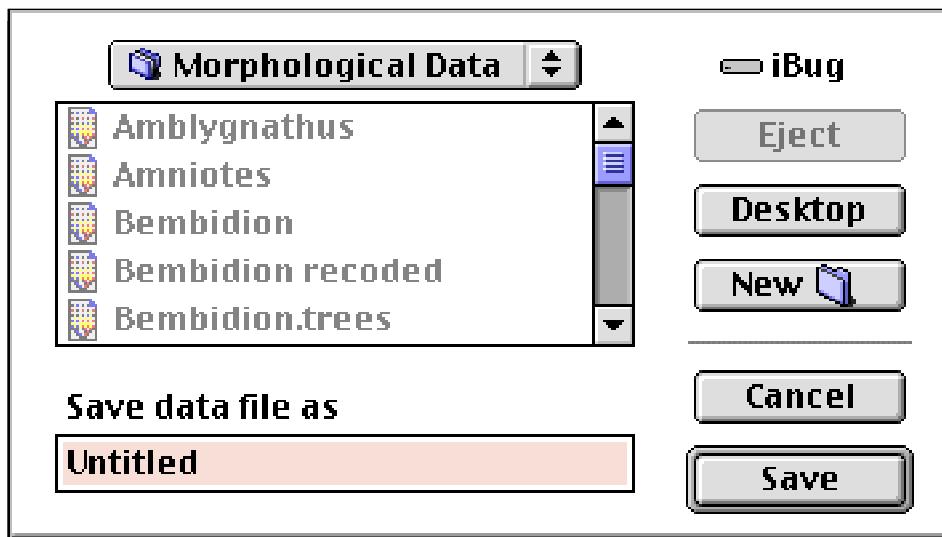
NOTE: *If you hold down the Option key while the file is being read, you will be taken automatically to the data editor, even if the file was last saved with the tree window on the screen. If you hold down the Command (⌘) key, you will be taken to the tree window. MacClade checks to see if a key is held down when the cursors change as it is reading the file.*

You can open only one file at a time. If you already have a file open, the **Open File** and **New File** menu items will be dimmed.

As you must open a data file before you open a tree file, MacClade will not allow you to open a tree file first.

Saving files to disk

To save the current data file choose **Save File** from the **File** menu. To save the current data file under a different name (while leaving the last-saved copy on disk under the old name), or to save a brand new file, choose **Save File As** from the **File** menu.



The **Save File As** dialog box

(The dialog box you see may vary depending upon the MacOS System version you are using.)

To save the current tree file, choose **Save Tree File** from the **Trees** menu. To save the currently stored trees to a tree file under a different name, choose **Save Tree File As** from the **Trees** menu.

MacClade maintains the data, including the data matrix, user-defined types, type sets, weight sets, trees, and character names in the computer's memory (RAM). Changes you make to these elements (e.g., by storing another type set with the **Store Type Set** command) are maintained in memory and are written to disk *only* when you issue an explicit **Save File** or **Save Tree File** command. Even when you edit the data matrix, the changes are not written to disk before you move to the tree window; they are only held in memory. When you do give a **Save** command, all of the changes you have made and items you have stored are saved to the disk.

To protect your work from system crashes and disk failures, save your file frequently and keep backup copies of your files in more than one place.

NOTE: If you hold down the Option key and choose **Save Stationery As** from the **File** menu, then MacClade will save the data file as a stationery pad (see "[Starting MacClade](#)" on page 113).

The format of MacClade files

Character data, trees, and other information of interest are stored in a MacClade data file or tree file in a very particular format. Generally the MacClade user is not concerned with the internal structure of the data files because MacClade digests this information and presents it to the user in the form of the data matrix in the data editor, a tree, and so on. When you save a file containing these objects, MacClade stores on the disk not the objects themselves, but a textual description of them. The format of this textual descrip-

tion is called the NEXUS format.

The NEXUS format was designed to make sharing of data between programs as easy and flexible as possible (Maddison, Swofford, and Maddison, 1997). Swofford's PAUP version 3.0 (1991) and PAUP* version 4.0 (2000) uses the NEXUS format, and with a few exceptions, MacClade's data and tree files are compatible with PAUP. (See [Chapter 26, "Using MacClade and PAUP* Together"](#).)

Details about the NEXUS format of relevance to MacClade users are provided in [Chapter 7](#).

Saving files in other formats

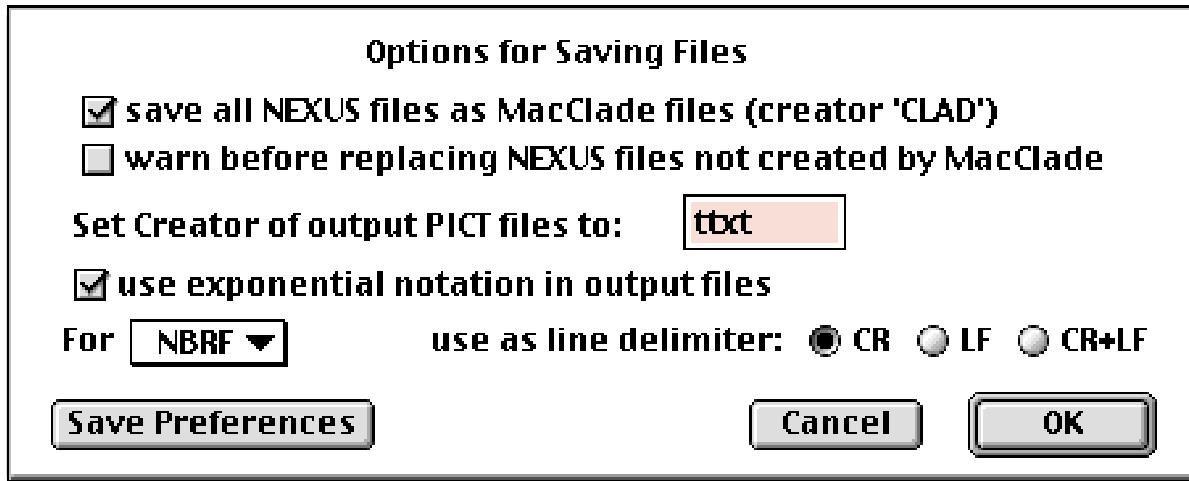
Using the **Export File** submenu, MacClade can write data files in formats other than NEXUS. For instance, MacClade will write HENNIG86 and PHYLIP files so that those programs can be used alongside MacClade. A tab-delimited table representing the data matrix can be written as a simple table file. A text file of descriptions can also be written. These exporting features are described in [Chapter 8](#).

Tree files can also be exported to HENNIG86 or PHYLIP formats using the **Export Tree File** submenu. This feature is described in [Chapter 8](#) and [Chapter 17](#).

Results from analyses can also be saved as files, such as graphics files containing a picture of the tree or a chart ([Chapter 25](#)), or text files storing the information contained in a chart ([Chapter 20](#)), or the results from the character correlation test ([Chapter 22](#)).

Altering the way MacClade writes files

If you want to change the way MacClade writes files, you can use the items in the **Options for Saving** submenu of the **File** menu to make a few changes. Of the two items in this submenu, the **NEXUS Format** item allows you to change the manner in which NEXUS files are formatted; details are given in [Chapter 7](#). The **Other Options** item produces a dialog box in which you can alter the way that MacClade saves a variety of files.



We will describe only the first two options here; the others are discussed elsewhere. Setting the creator of PICT files is discussed in [Chapter 25](#), use of exponential notation is described in [Chapter 20](#), and the export line delimiter is detailed in [Chapter 8](#).

On the Macintosh, files are marked as having been created by an application. Thus, if you create a new file

in MacClade, and save it to disk, MacClade will stamp its signature onto the file (by setting its creator to "CLAD"). The file then "belongs" to MacClade, it will have a MacClade data file icon, and if you double-click on it in the Finder, MacClade will be opened. If you ask MacClade to save all NEXUS files as MacClade files, then any file that MacClade saves using the **Save File** or **Save File As** commands will have their creator set to "CLAD" (and thus they will have MacClade icons), even if MacClade was not the original creator of the file. For example, if you had a file created in PAUP*, displaying a PAUP* icon, then you read that file into MacClade and saved it, MacClade would replace PAUP* as the creator of the file. If this option is not checked, however, and MacClade was not the original creator of the file, then MacClade will leave the creator as is, if you choose **Save File**. Thus, if a PAUP* file is read in and saved in MacClade, its creator would remain PAUP*. This will not affect the text of the file as written by MacClade.

Some elements of a data file are ignored by MacClade (such as some types of file comments) and omitted when MacClade rewrites the file. As some of these elements may be valuable to you, you can ask MacClade to warn you before it replaces any files that are not of creator "CLAD".

Reverting to last saved version

To revert to the last-saved version of a file and discard any subsequent changes, choose **Revert to Saved** from the **File** menu. This is not available if MacClade considers the file to be unsaved (e.g., a new file).

Closing data files

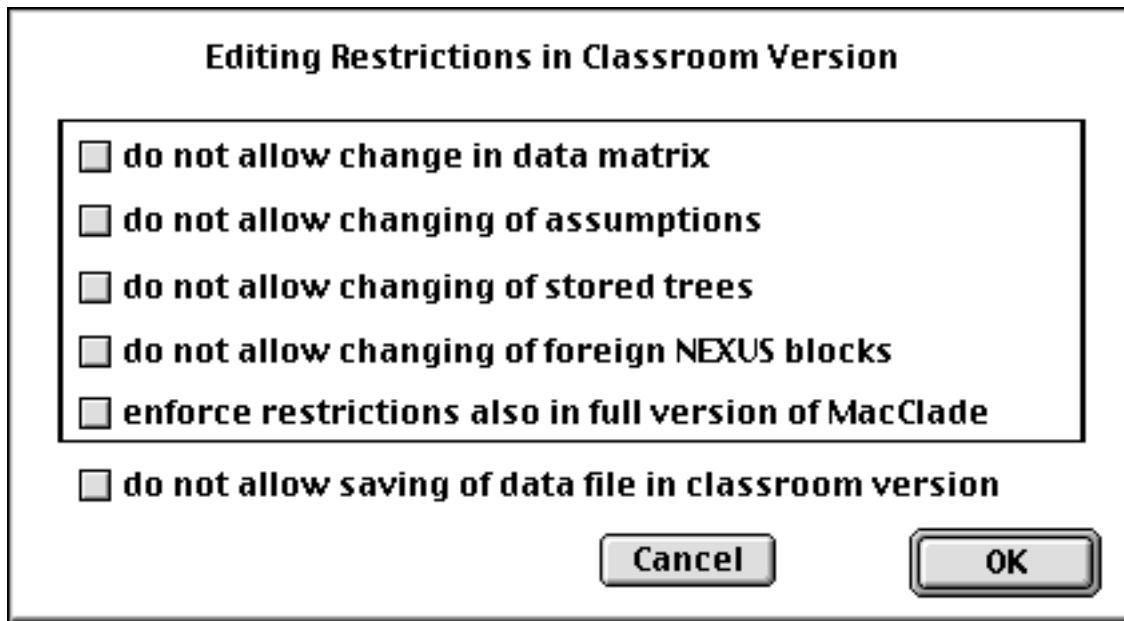
To close a file, choose **Close File** from the **File** menu. If you have made changes (such as editing data, rearranging the tree, changing assumptions, and so on) since the last time you saved the file, MacClade will ask if you want to save the changes before closing the file. If you press the Yes button, then MacClade will save changes and close the file; if you press No, MacClade will not save the changes and close the file; if you press Cancel, MacClade will not save the changes and not close the file.

Closing a window (e.g., by clicking on the go-away box of MacClade's tree window or data editor) does not necessarily close the file. If other windows belonging to the file (such as the character list window) remain open, the file will not close. MacClade closes the file when its last window is closed, or if the user chooses **Close File** (or **Quit**) from the **Edit** menu. Conversely, choosing **Close File** does not simply close the frontmost window; it also closes the file.

Editing restrictions in the classroom version

The classroom version of MacClade (on the CD, in the "MacClade 4 Classroom Version" folder) is identical to the normal version, with one exception: it has limited editing capabilities with data files that are so designated.

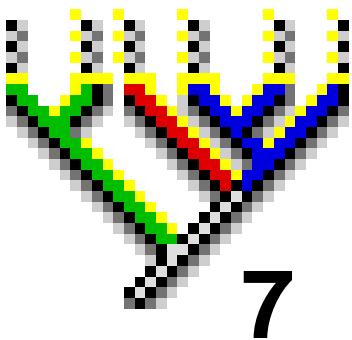
To specify that some changes for a data file are not allowed when examining it with the classroom version, you will need to open the file in the full, normal version of MacClade. Then choose **Editing Restrictions** from the **Edit** menu, and you will be presented with the following dialog box:



If you select the first option, to not allow changes in the data matrix, and then save the file, users of the classroom version of MacClade will then not be allowed to make any changes to the data themselves. In a similar fashion you may prevent changing of the assumptions, stored trees, and NEXUS blocks that are foreign (that is, those not understood by MacClade). For any of these options, if you also choose "enforce restrictions also in full version of MacClade", then even users of the full version of MacClade will not be allowed to alter those elements (unless, of course, they open the Editing Restrictions dialog box and allow those changes!).

In addition, you can prevent users of the classroom version from saving the data file by checking the option "do not allow saving of data file in classroom version". This does not prevent users of the full version from saving the file, only users of the classroom version.

Thus, for each file, you use the full version of MacClade to set the editing restrictions that will apply when using the classroom version (and, if so designated, the full version). Restrictions can vary from file to file.



7

NEXUS FILES AND BLOCKS

This chapter describes the NEXUS file format used by MacClade and other programs such as PAUP*. The file format is flexible and modular, and was described by Maddison, Swofford, and Maddison (1997). A brief description of the format, and of how MacClade supports editing of NEXUS files, is given here.

The NEXUS format

For basic MacClade use, you don't have to learn the NEXUS file format, as MacClade saves the information from the data editor, trees, type definitions, type sets, weights sets, character and state names, and so on, and writes it as a NEXUS formatted file. Space does not permit more than a brief outline of that format here. A full description is given by Maddison, Swofford, and Maddison (1997).

A NEXUS file begins with "#NEXUS", with the remainder of the file being organized into **blocks** of information. A small file with three taxa and one character might contain three blocks:

```
#NEXUS

BEGIN TAXA;
  DIMENSIONS NTAX=3;
  TAXLABELS
    taxon_1
    taxon_2
    taxon_3;
END;

BEGIN CHARACTERS;
  DIMENSIONS NCHAR=1;
  FORMAT MISSING=? GAP=- ;
  MATRIX
    taxon_1  0
    taxon_2  1
    taxon_3  1
  ;
END;

BEGIN ASSUMPTIONS;
  OPTIONS DEFTYPE=unord PolyTcount=MINSTEPS ;
END;
```

Each block is framed by a "BEGIN" statement and an "END;" statement. The "BEGIN" statement includes a token that is the type of the block. (In this example, there is a TAXA block, a CHARACTERS block, and an ASSUMPTIONS block.)

Within each block there is a series of commands, each command beginning with the command name, and ending with a semicolon (";"). In the following CHARACTERS block,

```
BEGIN CHARACTERS;
  DIMENSIONS NCHAR=1;
  FORMAT MISSING=? GAP=- ;
  MATRIX
    taxon_1  0
    taxon_2  1
    taxon_3  1
  ;
END;
```

the three commands are DIMENSIONS, FORMAT, and MATRIX.

The words of a NEXUS file consist of tokens. A token is a string of characters surrounded by whitespace (that is, blanks, tabs, carriage returns, or line feeds) or punctuation ([] () { } = , ; : - + \ / * " ^ ~ ^ < >). Tokens cannot contain any NEXUS punctuation or whitespace, or if they do, the token must be surrounded by single quotes (e.g., 'A token', 'Species 1; west'); any contained single quotes should be duplicated (e.g., 'Jane's Wren').

WARNING: You can easily see a NEXUS file directly by opening it with PAUP*'s editor, or by viewing it in a word-processing program. If you type in or modify the raw text file yourself using a word processor, PAUP*, or other program, and try to read it into MacClade, make sure you know what you are doing (see Maddison, Swofford, and Maddison, 1997), or you may see some spectacular crashes. Because MacClade should write perfect NEXUS files, it expects perfect NEXUS files in return. It thus does not perform extensive error checking on the files it reads in. We recommend that you let MacClade format the NEXUS file for you. If you edit NEXUS files using a word processor, the file must be saved as "Text Only" or MacClade will not even know the file exists. Also, be forewarned that any pictures you have in the data file **will be discarded** if you edit and save a MacClade file in most word-processing programs! (See "[Annotating cells in the data matrix](#)" in [Chapter 24](#).)

Processed and foreign NEXUS blocks

There are many other types of blocks that might be included in a NEXUS file. The types of blocks that MacClade can read and extract useful information out of are:

| | | |
|------------|-------------|----------|
| TAXA | ASSUMPTIONS | CODONS |
| CHARACTERS | SETS | NOTES |
| DATA | TREES | MACCLADE |

These are the *processed* blocks. MacClade maintains control over the processed blocks, and when you ask MacClade to save a file, it creates and formats the processed blocks for you.

If MacClade reads a file with other blocks, it will read and store these other blocks, but it won't be able to extract useful information out of them. You can edit the content of these *foreign* (i.e., those unfamiliar to MacClade) blocks as described later in this chapter.

If MacClade encounters multiple blocks of a sort that it understands (e.g., two TREES blocks), it will usually not process all of the blocks, and will treat all but one of them as foreign. In general, it will process only the first block of each sort, unless the first block is LINKed to an unprocessed block, in which case it will process the first block that is LINKed to a processed block. The only exception is that two CHARACTERS blocks will be processed if one has discrete data and the other continuous data. As of this writing (September 2000), files with LINKed blocks are very rare.

Cursors displayed while reading NEXUS files

Organismal cursors displayed as the data file is being read indicate which part of the file is being read. This can be handy if MacClade crashes while reading in a file, as you will know what part of the file MacClade was processing when it crashes. For the curious, and for debugging purposes, the correspondence between cursor and file is:

| Cursor | Organism | File |
|--------|----------------|---|
| | amoeba | initial commands in DATA, TAXA, or CHARACTERS blocks |
| | ground beetle | first half of matrix |
| | jumping spider | second half of matrix |
| | flower | various calculations |
| | mushrooms | TAXA block |
| | DNA | CODONS block |
| | owlfly | CHARACTERS block for continuous data |
| | trilobite | USERTYPES (ASSUMPTIONS block) |
| | anemone | TYPESETS (ASSUMPTIONS block) |
| | mouse | WTSETS (ASSUMPTIONS block) |
| | flatworm | EXSETS (ASSUMPTIONS block) |
| | bacteriophage | SETS block |
| | tree | TREES block |
| | starfish | NOTES block |
| | scallop | MACCLADE block |
| | fish | information foreign to MacClade, including foreign blocks |

Changing the way MacClade writes processed NEXUS blocks

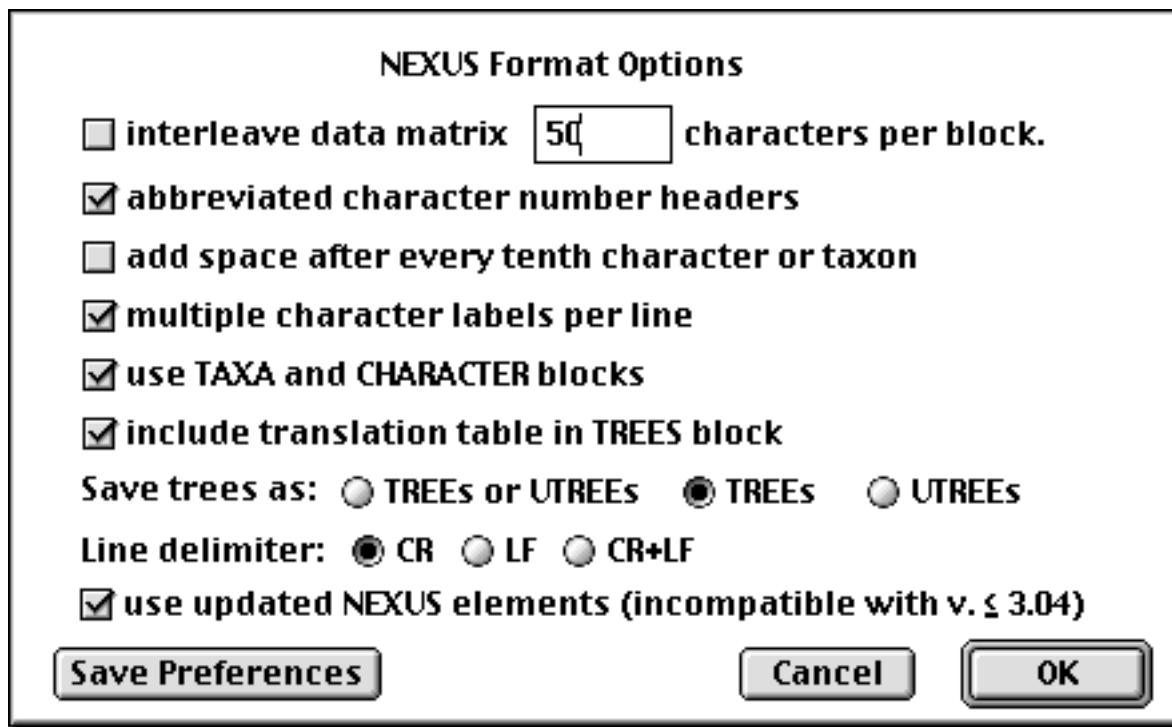
MacClade controls the format of processed blocks; however, there are some options that give you some choice about the format of some elements of these processed blocks.

Imagine a DNA sequence matrix written by MacClade consisting of 80 sites for five taxa. If the match-first facility is turned on in the data editor (["Matching first taxon" on page 258](#)), MacClade will write the data matrix in the file something like this:

```
[          10      20      30      40      50      60      70      80]
[          .      .      .      .      .      .      .      .]

Hylobates  AAGCTTTACAGGTGCAACCGTCCCTCATATAATGCCAACGGACTAACCTCTCCCTGCTATTCTGCCTTGCAAACCTCAAAC [80]
Homo       ....C..C..C...GT.A.T.....R..T..A..C..AT.A.....A.....[80]
Pan        ....C..C..C...TTA.....T..A..C..AT..AT.....A.....T. [80]
Gorilla    ....C..C..C...GTT..T..T....T.....T..A..A..AT..AT.....A.....[80]
Pongo      ....C..C..C.....AC.....G..T.....T....C..A..C.....A..G.....A.....[80]
```

You can control some aspects of this by changing the display in the data editor (e.g., transposing the matrix, choosing not to use the match-first facility, and so on). Some changes in the editor will also affect how MacClade writes the NEXUS file. A few other aspects can be controlled using the **NEXUS Format** menu item of the **Options for Saving** submenu in the **File** menu:



Turning on the interleave option in the **NEXUS Format** dialog box, and setting for 40 characters per block, will cause MacClade to write the matrix in more than one block:

```
[          10      20      30      40]
[       .       .       .       .]

Hylobates  AAGCTTTACAGGTGCAACCGTCCCTCATAATGCCAACGGA  [40]
Homo      ....C..C..C...GT.A.T.....R  [40]
Pan       ....C..C..C....TTA.....      [40]
Gorilla   ....C..C..C...GTT..T..T....T.....      [40]
Pongo     ....C..C..C.....AC.....G..T.....T....  [40]

[          50      60      70      80]
[       .       .       .       .]

Hylobates  CTAACCTCTTCCCTGCTATTCTGCCTTGCAAACCTCAAAC  [80]
Homo      ..T..A..C..AT.A.....A.....      [80]
Pan       ..T..A..C..AT.AT.....A.....T.  [80]
Gorilla   ..T..A..A..AT.AT.....A.....      [80]
Pongo     ..C..A..C.....A..G.....A.....      [80]
```

In typical files written by MacClade, the character number is written above every tenth character. This is an abbreviated character number header. It is placed within square brackets, and so has no impact on MacClade's or PAUP*'s reading of the file (as elements in square brackets are ignored), but is useful to help you look at the file. If abbreviated headers are turned off in the **NEXUS Format** dialog box, then character numbers will be written above each character:

```
[          1      2      3      4      5      6      7      8]
[ 1234567890123456789012345678901234567890123456789012345678901234567890]

Hylobates  AAGCTTTACAGGTGCAACCGTCCCTCATAATGCCAACGGA  [80]
Homo      ....C..C..C...GT.A.T.....R..T..A..C..AT.A.....A.....  [80]
Pan       ....C..C..C....TTA.....T..A..C..AT.AT.....A.....T.  [80]
Gorilla   ....C..C..C...GTT..T..T....T..A..A..AT.AT.....A.....  [80]
Pongo     ....C..C..C.....AC.....G..T.....T..C..A..C.....A..G.....A.....  [80]
```

Headers are written only for molecular sequence data and for standard or extended standard data in which no entries are polymorphic or partially uncertain.

If you request "add space after every tenth character or taxon", then MacClade will add a blank column after every tenth character (or taxon for a transposed matrix):

```
[          1      2      3      4]
[ 1234567890 1234567890 1234567890 1234567890]

Hylobates  AAGCTTTACA GGTGCAACCG TCCTCATAAT CGCCCACGGA  [40]
Homo      ....C..C ..C...GT.A ..T..... .....R  [40]
Pan       ....C..C ..C....TTA .....      [40]
Gorilla   ....C..C ..C...GTT..T..T....T.....      [40]
Pongo     ....C..C ..C.....A C.....G..T.....T...  [40]
```

If there are characters named in the matrix, and the matrix is not transposed, then MacClade will by default write up to 10 character labels on one line:

```
CHARSTATELABELS
 1 eye_color, 2 head_size, 3 number_of_toes
 ;
```

If you turn off "multiple character labels per line", then each character label will be placed on a separate line:

```
CHARSTATELABELS
 1 eye_color,
 2 head_size,
 3 number_of_toes
 ;
```

By default, MacClade 4 writes the discrete data matrix in separate TAXA and CHARACTERS blocks, not a single DATA block as in version 3. If you uncheck the "use TAXA and CHARACTERS blocks" in the **NEXUS Format** dialog box, then MacClade will write instead a single DATA block. Use of a single DATA block will make the program compatible with earlier versions of PAUP (version 3.0r and before), but less compatible with other programs.

By default, MacClade writes TREES blocks with a translation table; you can ask MacClade to omit the translation table by unchecking the appropriate box in the **NEXUS Format** dialog.

As MacClade treats all trees as rooted trees, by default it labels all trees it writes as rooted TREES. Users will notice that PAUP*, on reading a MacClade file with all trees saved as TREES, requests to unroot these trees whenever no directed transformation types are in use. If you want MacClade to write unrooted UTREES when no directed transformation types are in use, and TREES when they are, then select the "TREES or UTREES" option. With this option, PAUP* will not complain, but the user should be aware that the rooting as shown by MacClade may be lost in the process, and that PAUP* may substitute, in a sense, an alternative root. In most respects PAUP*'s UTREES are not rooted, except that something like a root is arbitrarily chosen to be between the first taxon in the matrix and the rest (if no outgroups are designated). This root position will influence both ACCTRAN/DELTRAN reconstructions and any subsequent true root that is designated for the trees if directed transformation types are reapplied (see [Chapter 26](#)). If you select the UTREES option, MacClade will always write UTREES.

By default, MacClade writes carriage returns (CR) at the end of each line of the file. However, carriage returns may not be the appropriate end-of-line marker if the file is to be moved to some other computer system, such as an MS-DOS machine, for which both a carriage return and line feed (LF) may be the most appropriate line delimiters (see ["Line delimiter" on page 140](#)). You can choose whether MacClade writes carriage returns, line feeds, or both carriage returns and line feeds.

NEXUS block management

MacClade looks after the creation of all of the NEXUS blocks containing information it processes. For example, when you create a data file in MacClade, and ask MacClade to save it, it writes into the file a block (perhaps called CHARACTERS) that contains the data matrix, names of characters, and so forth, written in NEXUS commands. If the only program you ever use is MacClade, then you can let MacClade do all of the reading and writing of NEXUS blocks for you, without ever worrying about them. However, if you want to use the file in other programs, such as PAUP*, it can be handy or necessary to learn how to write NEXUS blocks yourself.

List of blocks in the file

To examine a list of NEXUS blocks in the data file, choose **NEXUS Block List** under the **NEXUS Blocks** submenu of the **Edit** menu. You will be presented with a window:

| Block | MacClade | Foreign Lines | Will Write |
|-----------------------|----------|---------------|------------|
| 1 MacCladeStart | • | | |
| 2 Taxa | • | | • |
| 3 Data | • | | |
| 4 Characters | • | | • |
| 5 Characters (conts.) | • | | |
| 6 Codons | • | | |
| 7 Sets | • | | |
| 8 Assumptions | • | | • |
| 9 Trees | • | | |
| 10 Notes | • | | |
| 11 MacClade | • | | • |

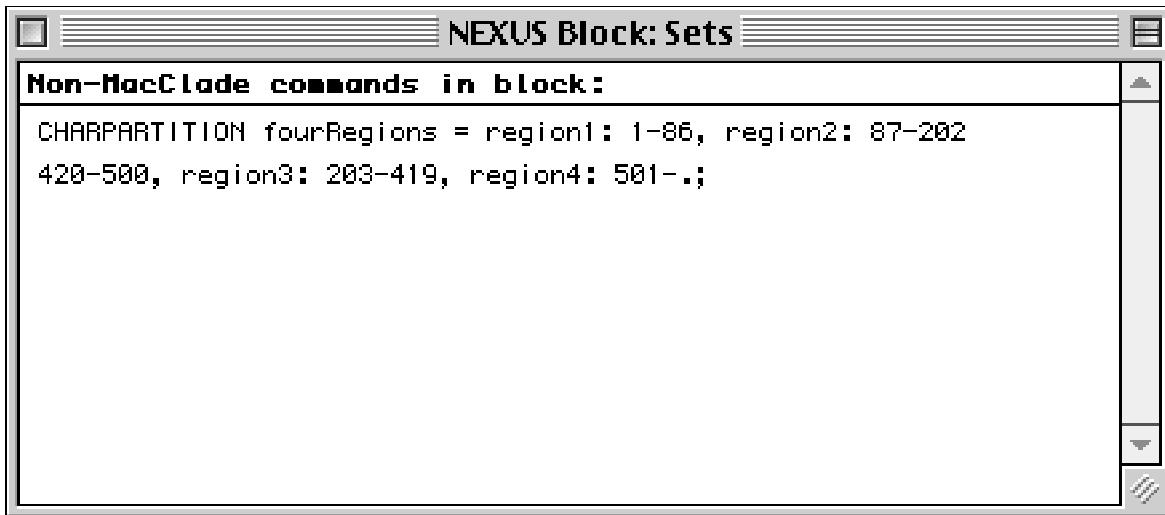
This list shows all of the blocks MacClade might process or create, including ones that are not present in the current file, as well as any blocks foreign to MacClade. A dot in the MacClade column indicates that this is a block MacClade would typically process if encountered in a file. The Foreign Lines column indicates if there are any foreign lines (commands not processed by MacClade) in a processed block. If MacClade will include a particular block within a file when the file is written, there will be a dot in the "Will Write" column. Lines with gray text are blocks to which you can not add or edit foreign lines from within MacClade.

Adding foreign commands to a processed NEXUS block

MacClade allows you to add foreign commands to some of the blocks it processes (including the CODONS, SETS, ASSUMPTIONS, TREES, and NOTES blocks). In the NEXUS block list window, some of the blocks listed are grayed, some are black. You can add foreign commands only to those blocks shown in black. To do this, double click on the row for a block. MacClade will present you with a text window in which you can enter NEXUS commands to be added at the end of those blocks when MacClade saves the file.

For example, imagine that you wish to conduct a likelihood analysis in PAUP* in which a gene sequence is divided into four regions, with different rates of evolution in each of the four regions. To do this, you would need to create a character partition that will define the boundaries of the four regions. As MacClade does not create character partitions itself, you will need to create the CHARPARTITION command yourself, and store it in the SETS block. To do this, double click on the SETS block in the NEXUS block list

window, and you will be presented with a window in which you can enter a CHARPARTITION command:

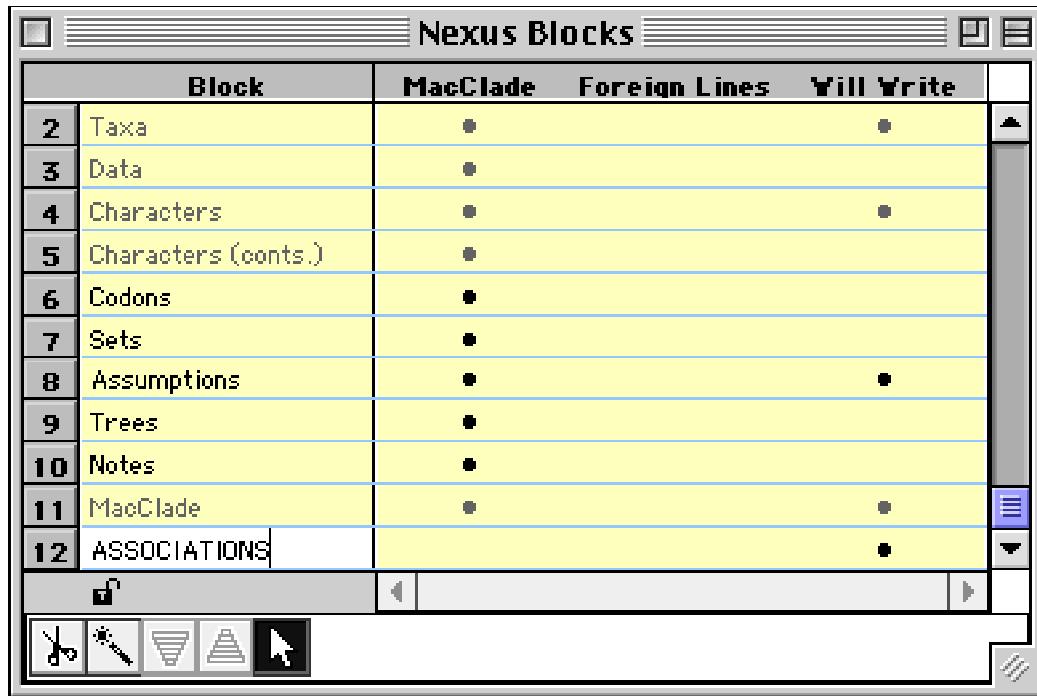


When MacClade saves the file, this CHARPARTITION command will be saved at the end of the SETS block, and will be available to PAUP* once PAUP* rereads the file.

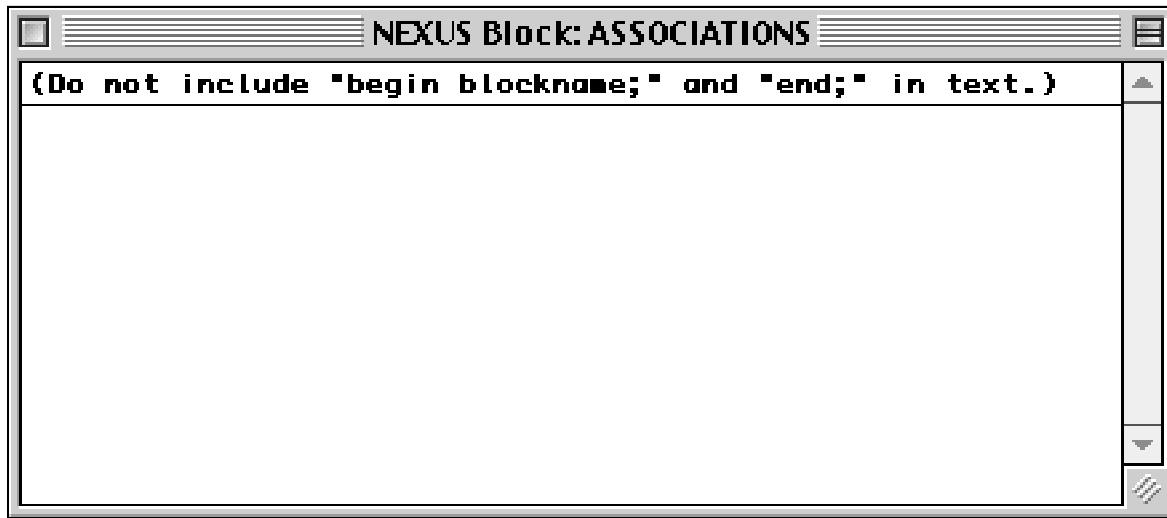
Creating a new NEXUS block

MacClade 4 allows you to create additional NEXUS blocks and edit them from within MacClade itself. To create a new NEXUS block, choose **New NEXUS Block** from the **NEXUS Blocks** submenu of the **Edit** menu. You will be presented with the NEXUS block list window, showing you all of the NEXUS blocks in the file. This will include the new, untitled block you just created. Type in the appropriate name for the block. You can enter text into the block by double-clicking on the block's number in this list window, or by selecting the block's row and choosing **Edit NEXUS block** from the **NEXUS Blocks** submenu.

For example, imagine that you have a program that requires an ASSOCIATIONS block. To create one, choose **New Block** from the **NEXUS Blocks** submenu of the **Edit** menu, and name the new block "ASSOCIATIONS":



Double-click on the ASSOCIATIONS block's number and you will be presented with an empty editor window:



You can now enter the commands for the ASSOCIATIONS block in this editor. Don't enter the two NEXUS commands that form the boundaries to the block; that is, don't enter "BEGIN ASSOCIATIONS;" and "END;" as these will be written into the file automatically by MacClade.

The **Open PAUP Block** command of the **NEXUS Blocks** submenu will open the first PAUP block in the file, and, if none exists, create one. This allows you to add quickly PAUP* commands to a file.

Changing the order of NEXUS blocks

To change the order of the foreign blocks in a NEXUS file, grab the number of the block in the NEXUS block list window, and move the block to its desired position. For example, you might wish to move the ASSOCIATIONS block to just after the TREES block:

| | Block | MacClade | Foreign Lines | Will Write |
|----|---------------------|----------|---------------|------------|
| 2 | Taxa | • | | • |
| 3 | Data | • | | |
| 4 | Characters | • | | • |
| 5 | Characters (conts.) | • | | |
| 6 | Codons | • | | |
| 7 | Sets | • | | |
| 8 | Assumptions | • | | • |
| 9 | Trees | • | | |
| 10 | ASSOCIATIONS | | | • |
| 11 | Notes | • | | |
| 12 | MacClade | • | | • |

MacClade will not let you move processed blocks (those marked with a dot in the MacClade column).

Special features for PAUP blocks

Because of the frequent use of MacClade with PAUP*, there are some special features added to MacClade to aid in the creation of PAUP blocks.

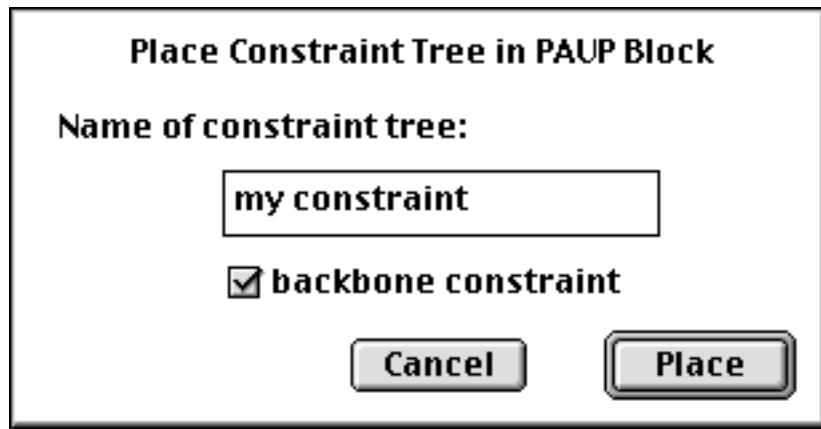
The **Open PAUP Block** command in the **NEXUS Blocks** submenu of the **Edit** Menu will open a window allowing you to edit the first PAUP block in the file; if a PAUP block is not present in the file, MacClade will create one.

MacClade can insert automatically some commands specifically for use in PAUP blocks. This can be done using items in the **Place PAUP Command** submenu in the **Edit** menu.

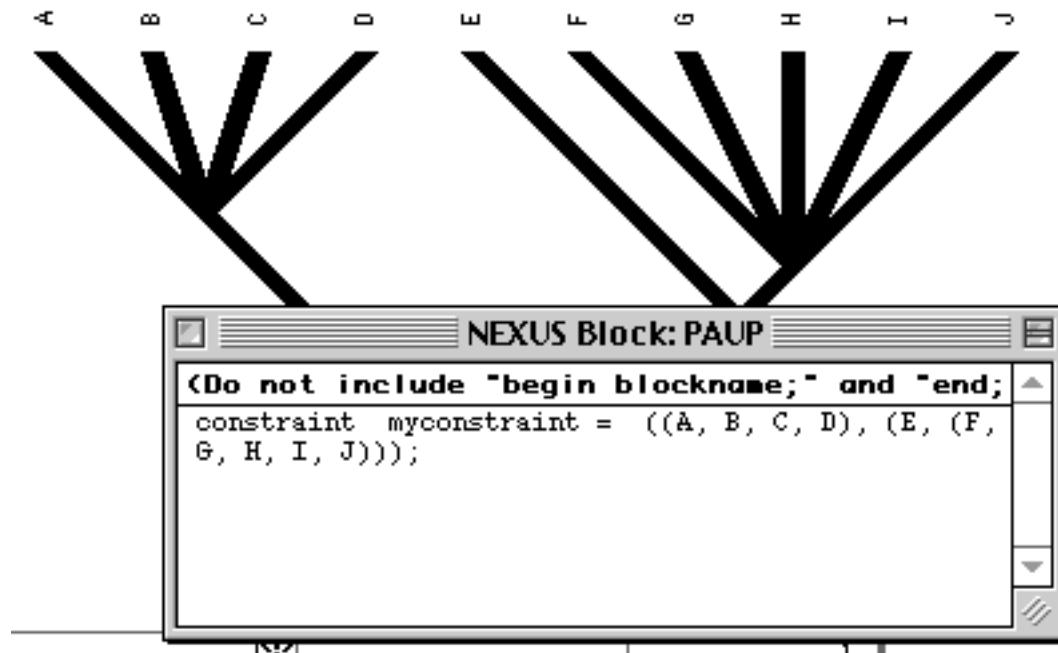
Constraint trees

If you wish to have a constraint tree available to PAUP*, either for searching for filtering, you can quickly generate the relevant command in MacClade. To do this, you will need to be in MacClade's tree window..

Arrange the tree as you wish for the constraint tree. Now choose **Constraint Tree...** from the **Place PAUP Command** submenu, and you will be presented with a dialog box allowing you to name the constraint:



In addition to naming the constraint tree, you can also choose whether or not the constraint will be saved as a backbone constraint if some taxa have been excluded from the tree. The PAUP* manual describes backbone constraints. Pressing Place in this dialog box will place the CONSTRAINT command in the PAUP block:

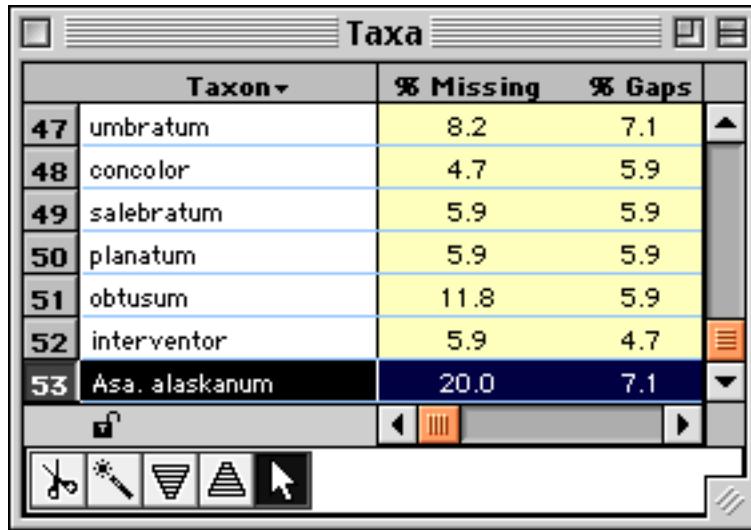


When you now save the file, MacClade will write into the file a PAUP block containing your constraint tree, for use by PAUP*.

Defining outgroups

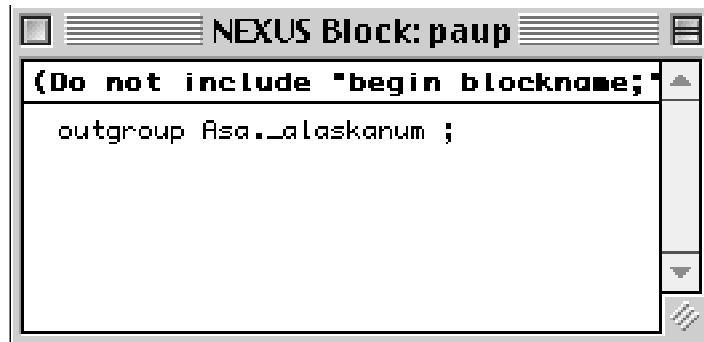
The **Outgroup** menu item of the **Place PAUP Command** submenu will place an OUTGROUP command into the PAUP block. To use this, you will first need to open the taxon list window, and select the taxa

(["Selecting objects" on page 163](#)) you wish to be included in the outgroup. For example, by selecting "Asa. alaskanum" in the Taxa list window:



| Taxon | % Missing | % Gaps |
|-------------------|-----------|--------|
| 47 umbratum | 8.2 | 7.1 |
| 48 concolor | 4.7 | 5.9 |
| 49 salebratum | 5.9 | 5.9 |
| 50 planatum | 5.9 | 5.9 |
| 51 obtusum | 11.8 | 5.9 |
| 52 interventor | 5.9 | 4.7 |
| 53 Asa. alaskanum | 20.0 | 7.1 |

and then choosing **Outgroup** from the **Place PAUP Command** submenu, a command specifying *Asa. alaskanum* as outgroup will be placed in the PAUP block:

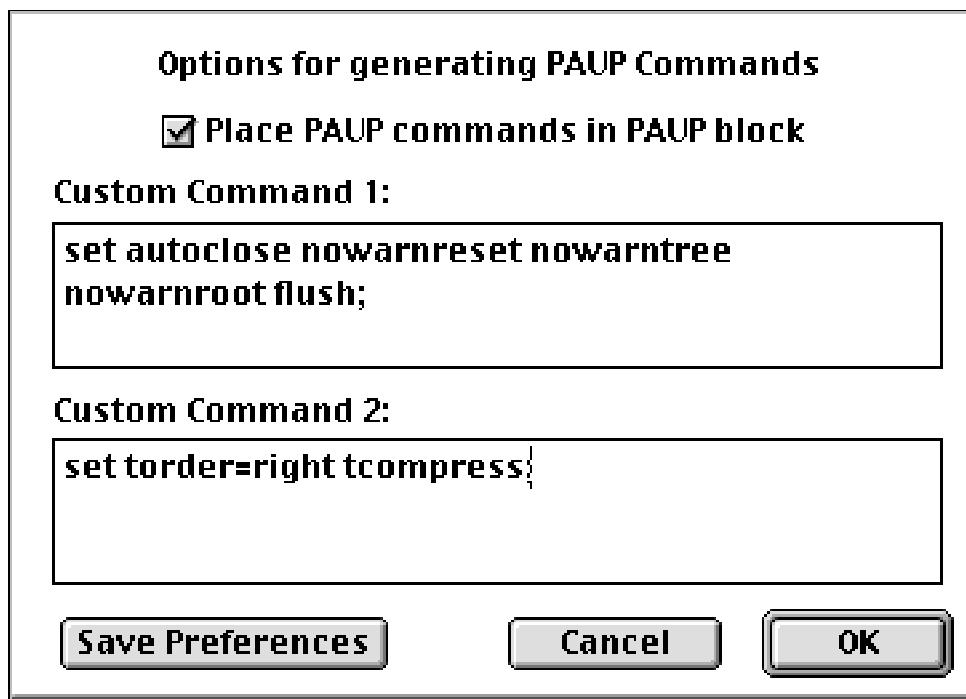


```
(Do not include "begin blockname;"'
outgroup Asa.alaskanum ;
```

Custom commands

In addition to the CONSTRAINT and OUTGROUP commands, MacClade will automatically place in a PAUP block two custom commands, of your choosing. These are entered into the PAUP block when you choose the **Custom 1** or **Custom 2** commands of the **Place PAUP Command** submenu. To edit the commands that are placed when you choose these menu items, use the **Options...** command in the **Place PAUP Command** submenu. In the dialog box presented, you will be able to edit the custom commands (and

save them as your desired commands for future uses of MacClade by pressing the Save Preferences button):



This dialog box also allows you to choose whether the **Place PAUP Command** submenu automatically places the chosen command in the PAUP block (the default) or into the clipboard. (You can also switch between these two modes by holding down the Option key when choosing items from the **Place PAUP Command** submenu.)

Preparing a NEXUS format file by hand

If you have data already stored in a file format other than NEXUS, you may be able to convert it to NEXUS format by importing the file (see [Chapter 8](#)), then saving the file using the standard **Save As** dialog box. However, you may choose to edit the raw data file by hand to convert it to NEXUS format, perhaps because its current format is not one that MacClade can import.

If you want to convert your file to NEXUS format and you don't want to spend time learning the NEXUS format (Maddison, Swofford, and Maddison, 1997), one simple method is to create and save in MacClade a simple NEXUS data file of the same format of data as yours (for instance, DNA data). You can then open up this template file in a word processor. With the word processor, replace the data matrix in the NEXUS file by your own data matrix, and adjust the number of taxa and characters indicated (by adjusting the DIMENSIONS NTAX= NCHAR= line). Or, you can use the following small NEXUS file as a template for producing your own.

```

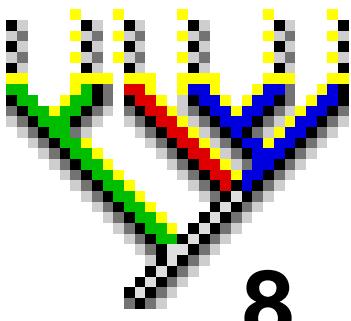
#NEXUS
      Replace this number
      by actual number of taxa
      ↓
BEGIN DATA;
DIMENSIONS NTAX= 2 NCHAR= 4 ;
FORMAT MISSING=? GAP=- DATATYPE= DNA ;
      Replace this number
      by actual number
      of characters
      ↑
MATRIX
      Replace this by
      RNA, PROTEIN,
      or STANDARD,
      as appropriate.
      ↑
Taxon1 ACG?
Taxon2 ACT?
;
END;
      Replace this by your data matrix
  
```

As long as your data matrix is not too strangely formatted, this may be a quick way to convert it by hand to NEXUS format. Be careful that taxon names do not include spaces or other illegal characters, or if they do, are properly protected by quotes (see the section "[The NEXUS format](#)", at the start of this chapter). Make sure, if you are using a word processor, to save the file as "Text Only". Some editors, such as those provided by PAUP* and BBEdit®.

Changes in the NEXUS format

The NEXUS file format changed slightly between MacClade versions 3.04 and 3.05. These changes were made in the course of preparing the formal description of the NEXUS format for publication. They primarily affect the genetic code and codon positions, continuous data, user-defined types and weights with real values, italicized taxon names, and character sets. Some of the new features of the file format will be disabled if you uncheck the "use updated NEXUS elements" in the **NEXUS Options** dialog box in the **Options for Saving** submenu. Use of these new features may make your file unreadable by early versions of MacClade 3.

In addition, the rules regarding punctuation changed between 3.04 and 3.05. The new rules dictate that +, ` [backquote], <, and > are punctuation, which will break a token in two; the old rules did not specify these as punctuation. If you want to force MacClade 4 to use the old rules when reading a file, you can do this by going to the **Save Preferences** dialog box in the File menu, check "old punctuation", and press the Save Preferences button.



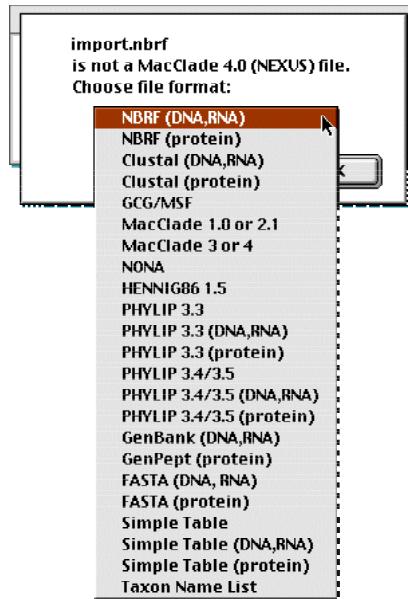
IMPORTING AND EXPORTING TEXT FILES

8

This chapter describes importing and exporting files from and to various formats used by other computer programs. Because MacClade and PAUP* use the same file format, no special importing and exporting between them is needed. For more details on general aspects of opening, closing, and saving files, see [Chapter 6](#).

Importing files

Although MacClade is designed to deal primarily with NEXUS format files, MacClade can read data files written in several other text formats (MacClade 1.0 and 2.1, PHYLIP, HENNIG86 version 1.5, NONA, NBRF, Clustal, GCG/MSF, FASTA, GenBank and GenPept, simple tables, and lists of taxon names). If you have a data matrix that is stored in one of these formats, you can import the data file into MacClade as follows. Open the file as you would any other data file (see [page 119](#) for details). MacClade will detect that the file is not in standard NEXUS format and present a dialog box with a pop-up menu from which you are to choose the file format. Make sure you choose the exact format:



For example, if your file is in NBRF format, and it contains DNA sequence data, then make sure you select the **NBRF (DNA, RNA)** menu item. For more details on importing files of foreign formats, see the appropriate sections, later in this chapter.

If you have data not stored in one of the formats that MacClade understands, you will have to use a word processor to manipulate the data file into an understandable format. It is relatively easy to format files for import into MacClade. Depending on your data and assumptions, you may find it easiest to convert to the NEXUS format for direct reading by MacClade ([page 137](#)), or you may find it easier to import the data into MacClade via some other format. If you have only the data matrix itself to import, remember that a text file

consisting of a matrix by itself may be directly importable by MacClade (discussed under "[Simple table files](#)" on page 145).

Importing data after manual editing of the files may be especially common for molecular sequence data acquired from other programs or sequence data banks. Two facts should be kept in mind. First, each sequence must be of the same length (except for NBRF and GenBank files, for which different sequences can be of different lengths). If you do not find MacClade's options for alignment suitable, you may find it convenient to have the sequences pre-aligned. To indicate gaps implied by alignment, use a hyphen, instead of blank spaces. Second, there may be restrictions on the names of the taxa. (Names with spaces may need to be treated in a special way, for instance.)

To import a tree file, choose **Open Tree File** from the **Trees** menu, select the file, and press the Open button. If MacClade cannot quickly determine the file format, it will present a dialog box asking for the format. MacClade understands tree files in NEXUS, PHYLIP, and NONA/HENNIG86 formats only.

An imported file will be opened as an Untitled document.

If MacClade's importing facilities cannot translate your data file, you may need to translate it by hand to NEXUS format (see [page 137](#)).

Exporting files

Use the **Export File** submenu to choose the format of export. On exporting files, MacClade checks to see if it can successfully write the current file in the requested format. If you ask to export a file to PHYLIP that has only binary characters, or that is of DNA or RNA or protein sequences, then MacClade will (usually) simply export the file. Similarly, files exported to HENNIG86 that have 10 or fewer states per character, with characters of unordered or ordered types, and no polymorphisms or partial uncertainty, will be written by MacClade without any warning. However, other data or assumptions may not be exportable (e.g., if a character is of a character-state-tree type with polymorphisms), and MacClade will warn you to that effect. In such a case, you will have to change your data or assumptions to a form compatible with the other program before requesting exportation.

Line delimiter

By default, MacClade will put a carriage return ("CR"; ASCII value 13) between each line in an exported file. But if you wish to move the file to a non-Macintosh computer, then a carriage return may not be the appropriate line delimiter. For example, if you export an NBRF file and want to move it to a UNIX machine, it may be more appropriate to have a line feed ("LF"; ASCII value 10) between lines. For a computer running Microsoft Windows, a carriage return plus line feed may be most appropriate. You can adjust the line delimiters used by MacClade for each of the exported file types in the **Other Options** dialog box in the **Options for Saving** submenu of the **File** menu. To change the export delimiter for, say, HENNIG86 files, choose **HENNIG86** from the pop-up menu on the left. Then click on the radio button on the right to specify the appropriate delimiter. You can ask MacClade to write carriage returns (CR), line feed (LF), or both carriage returns and line feeds.

The software you use to transfer files between computers may automatically translate the line delimiters. Before you adjust the line delimiters within MacClade, you should test out your system to see if your file transferral software automatically adjusts the delimiters. If your software already does this, then you might ignore this option in MacClade.

MacClade 1.0 and 2.1 files

MacClade 4 will import old MacClade files. MacClade will not export files into MacClade 1.0 or 2.1 format. MacClade 4 will *not* import or export files of some test versions of MacClade that were never released for general use (e.g., versions 2.65, 2.87, 2.97; see ["Future plans for MacClade" on page 20](#)).

MacClade is not always successful in importing MacClade 1.0 or 2.1 files if these files did not comply strictly with their intended file formats. MacClade 1.0 and 2.1 would accept files that violated their formats in minor ways. It is therefore possible to have a file that the old versions would accept, but that MacClade 4 will not. An example is the "Vertebrates" example file distributed with MacClade 2.1, which will not be read properly because one of the tree descriptions is lacking a necessary comma.

MacClade 3 files

As the file format used by MacClade 4 is essentially the same as that used by MacClade 3, for the most part no special importing is needed. Because of a slight change in the file format between versions 3.04 and 3.05, you may need to consider the information given in ["Changes in the NEXUS format" on page 138](#).

PAUP* files

With files created for PAUP* (Swofford, 1991, 2000), you don't have to import or export them in any special way, because MacClade 4, PAUP 3.0, and PAUP* 4.0 share the same NEXUS file format (see [Chapter 7](#)). However, there are some incompatibilities as indicated in [Chapter 26](#).

PHYLIP files

Importing

MacClade 4 can understand most PHYLIP 3.3, 3.4, and 3.5 files (Felsenstein, 1993); it can also import some files from earlier versions of PHYLIP. Because of the ambiguity in pre-PHYLIP 3.4 file formats (caused by the fact that different options in the file mean different things depending upon the PHYLIP program that processes the file), importing PHYLIP 3.3 files is fraught with peril, unless you remove all options, or pay close attention to the following notes.

The first line of the data file must contain only the number of taxa, number of characters, and letters for each of the options. Other numbers, such as required in version 3.3 of PHYLIP's RESTML, are not allowed.

For PHYLIP 3.3 files, MacClade processes options in the following ways:

| Option | Code | MacClade's action |
|------------|------|--|
| ALL | A | Must be removed |
| ANCSTATES | A | Ignores and skips over ancstates line |
| CAMIN | C | Must be removed |
| CATEGORIES | C | Ignores and skips over two categories line |
| CLIQUE | C | Must be removed |
| DOLLO | D | Sets all characters to Dollo |

| Option | Code | MacClade's action |
|--------------|------|---|
| FACTORS | F | Ignores and skips over factors line |
| FREQUENCIES | F | Ignores (molecular sequence data only) |
| GLOBAL | G | Ignores |
| HALF JACK | H | Ignores |
| JACKKNIFE | J | Must be removed |
| JUMBLE | J | Must be removed |
| LENGTH | L | Ignores |
| LOCAL | L | Ignores |
| MIXED | M | Sets character types, and stores type set |
| OUTGROUP | O | Ignores and skips over outgroup line |
| POLYMORPHISM | P | Ignores |
| RECONSIDER | R | Ignores |
| REPLICATE | R | Ignores |
| ROOTED | R | Ignores |
| SOKAL | S | Sets all characters to irreversible |
| THRESHOLD | T | Ignores and skips over line |
| TRANS/TRANS | T | Ignores and skips over line |
| USERTREES | U | Reads in trees |
| WEIGHTS | W | Sets character weights, and stores weight set |
| WRITE TREES | Y | Ignores |

Those options that are marked "**Must be removed**" indicate those options that must be removed from the data file before you import it into MacClade, or MacClade may crash on reading the file.

For PHYLIP 3.4 and 3.5 files, MacClade processes options in the following way:

| Option | Code | MacClade's action |
|------------|------|---|
| ANCSTATES | A | Ignores and skips over ancstates line |
| CATEGORIES | C | Ignores and skips over categories line |
| FACTORS | F | Ignores and skips over factors line |
| MIXED | M | Sets characters types, and stores type set |
| WEIGHTS | W | Sets character weights, and stores weight set |

For a nonmolecular matrix, MacClade will read "B", "b", "P", or "p" and set the state equal to "polymorphic for states 0 and 1". For molecular sequence data, it will read the standard IUPAC symbols appropriately.

MacClade presumes that you have allocated 10 characters for taxon names, and so presumes that data for the first character of a taxon begins at the 11th position on the line. In reading in a taxon name, any semicolon, ";", will be converted to an underscore, "_".

If trees follow the matrix, MacClade will read them.

Exporting

MacClade will write the following options, if applicable, into a PHYLIP data file:

| Option | Code | MacClade's action |
|---------|------|---------------------------------|
| MIXED | M | Writes currently active types |
| WEIGHTS | W | Writes currently active weights |

For version 3.3 PHYLIP files, MacClade will also include the following options, if relevant:

| Option | Code | MacClade's action |
|--------|------|---|
| DOLLO | D | Include if default type for file is Dollo |

MacClade will also include any trees in the current data file into the PHYLIP file (and add the Usertrees option for version 3.3 files).

If you choose to export a protein data file, be wary that a program like PHYLIP will not accept extra amino acid states, such as MacClade's states 1 and 2 for serine with the standard nuclear genetic code. If you do have such extra states in your matrix, you should recode them first using the **Recode** or **Search & Replace** commands before exporting the file.

Remember, if you ask to export a file using data or assumptions that MacClade considers incompatible with PHYLIP, MacClade will refuse, and you must revise your data or assumptions before exportation.

HENNIG86 files

Importing

MacClade can import HENNIG86 version 1.5 (Farris, 1988) data files and tree files. Because CLADOS (Nixon, 1992) uses Hennig86's data file format, MacClade can also import some files made for CLADOS. It will read in matrices in XREAD commands, trees in TREAD commands, assumptions in CCODE commands, and QUOTE commands. All other HENNIG86 commands are ignored. For MacClade to process trees, they must be in the format that HENNIG86 writes tree files (that is, fairly standard parenthesis notation). MacClade cannot understand trees in Hennig86's newer format that uses symbols such as []/\{\}. Elements in the title line of the XREAD command and in the QUOTE command are added to the File Notes window in MacClade.

Exporting

MacClade will export data and tree files in HENNIG86 version 1.5 format. It writes XREAD, TREAD, CCODE, and QUOTE commands. Included in the QUOTE command will be any text from MacClade's File Notes window. If any characters or states have names, MacClade will write these in the QUOTE command. In writing the matrix, MacClade will use "-" for gaps, "?" for missing data. All trees stored in the data file will be written into a TREAD command.

Remember, if you ask to export a file using data or assumptions that MacClade considers incompatible with HENNIG86 version 1.5, MacClade will refuse, and you must revise your data or assumptions before exportation.

NONA files

Importing

The file format used by NONA (Goloboff, 1999) is very similar to that of Hennig86. MacClade can read those elements of NONA files described in the previous section, as well as the DREAD command (for DNA data) and the CNAMES command (for character and state names).

Exporting

In exporting data and tree files in NONA format, MacClade will write the same commands that it does for Hennig86 files, as well as DREAD and CNAMES commands.

NBRF files

Importing

MacClade can read in National Biomedical Research Foundation (NBRF) format. These nucleotide or amino acid sequence files have, for each sequence, one line starting with ">", followed on that line by items that are irrelevant to MacClade. The next line contains the name of a sequence, followed by one or more lines of sequence, ending in "*":

```
>DL; taxon_1
a comment about taxon_1
AGGAAGTACGATCCGGCACA
CAACGGTTGTAGCACTTC*
>DL; taxon_2
a comment about taxon_2
TCGGATGTGTCAAGCGCTGG
TTGAGGTGATAATACTCTG*
>DL; taxon_3
a comment about taxon_3
TAAGGGTCCCATTATGTGA
GCCACATTAATTGTATTACG*
```

This file format is used by many programs that process molecular sequence data.

Exporting

MacClade also writes out files in National Biomedical Research Foundation format. This is available only for molecular sequence data.

If you choose to export a protein data file, be wary that a program like Jotun Hein's (1990) TreeAlign will

not accept extra amino acid states, such as MacClade's states 1 and 2 for serine with the standard nuclear genetic code. If you do have such extra states in your matrix, you should recode them first using the **Recode** or **Search & Replace** commands before exporting the file.

If you choose **NBRF** from the **Export File** submenu, MacClade will include any gaps in the file. In contrast, if you choose **NBRF no gaps**, MacClade will not write any gaps in the file.

CLUSTAL, GCG/MSF, FASTA, GenBank, and GenPept files

Importing

MacClade 4 can read the alignment (.aln) files created by Clustal V, W, and X, as well as files in GCG/MSF, FASTA, GenBank, and GenPept formats. Some programs produce files in these formats that vary slightly from the format's definition, and MacClade may have problems reading such files. If so, you may need to save them in one of the other file formats MacClade understands before importing.

Simple table files

Importing

MacClade can also read text files that are simple tables of character state data. The format for simple table files for importing is very specific: on each line must be all the data for one taxon, beginning with the taxon name, then a blank space or tab, then the state of the first character, state of second character and so on until the last character, which is followed by a carriage return. Blanks or tabs can be placed between states in the matrix, but they need not be there. The taxon name must have no spaces (use underscores to indicate spaces), unless it is surrounded by single quotes, and states must be listed as single-digit integers (0...9) and letters (A through R excluding I and O), or ACGTU if DNA or RNA data, or the appropriate IUPAC codes if protein data. The symbol "-" will be treated as a gap, "?" as missing data.

NOTE: Technically, the taxon name must be a single NEXUS token. See [Chapter 7](#) for details.

No polymorphisms or partial uncertainties are allowed. There can be comments (surrounded by []) before the matrix begins, and after it ends, but there can be nothing else in the data file.

Here is a simple table file that MacClade can read:

```
[can put comment here]
taxon_1      3 - 2 3 1 3 ? 0
taxon_2      0 0 0 1 2 0 1 2
taxon_3      2 2 3 ? 2 0 1 0
taxon_4      1 2 0 0 0 3 2 1
taxon_5      2 0 2 0 1 2 1 3
taxon_6      3 0 2 0 3 0 1 2
taxon_7      1 0 2 0 0 3 3 2
taxon_8      3 1 0 0 0 2 0 1
```

Exporting

MacClade allows you to export text files in simple tabular format, by choosing the **Simple Table** menu of the **Export File** submenu. You will be presented with a dialog box in which you choose the format of the table file:



Checking "include column titles" will cause MacClade to insert a title above each column in the matrix. Footnotes contained in your matrix can be included in the file by checking "include footnotes" in the dialog box. The choice of "character state legends" will write a list of the symbols used in the table and their state names on the left below each character name for transposed matrices, and below the matrix for nontransposed matrices. Part of a transposed table with character state legends looks something like this:

| | | | | | | | | | | |
|--------------|---|---|---|---|---|---|---|---|---|---|
| silver spots | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 |
|--------------|---|---|---|---|---|---|---|---|---|---|

0: absent
1: present

| | | | | | | | | | | | |
|---------------|---|---|---|---|---|---|---|---|---|---|---|
| int. 3 micro. | 0 | 0 | 0 | 1 | 1 | 1 | 3 | 3 | 2 | 3 | 3 |
|---------------|---|---|---|---|---|---|---|---|---|---|---|

0: absent
1: v. slight mirrors
2: slight mirrors
3: distinct mirrors

If you choose "state names in table", then MacClade will write full state names for character states if these are available, rather than single-character symbols.

If you choose "MacClade readable", then the table will be of a structure that will allow it to be imported back into MacClade. If this option is chosen, most others (such as "include column titles") are disallowed, as they would make the resulting file unreadable by MacClade.

If the interleave format is set in the **NEXUS Format** dialog under **Options for Saving** submenu in the **File** menu (see "[Changing the way MacClade writes processed NEXUS blocks](#)" on page 128), then the table will be interleaved (unless "MacClade readable" is checked).

Taxon name lists

Importing

If you wish to create a matrix from a list of taxa, MacClade will read in a list of taxa in the following format in a simple text file:

```
taxon 1  
taxon 2  
taxon 3  
taxon 4
```

and create a matrix with all of the taxa and one empty character. Each line of the file should have the name of one and only one taxon; the taxon names need not be formal NEXUS tokens.

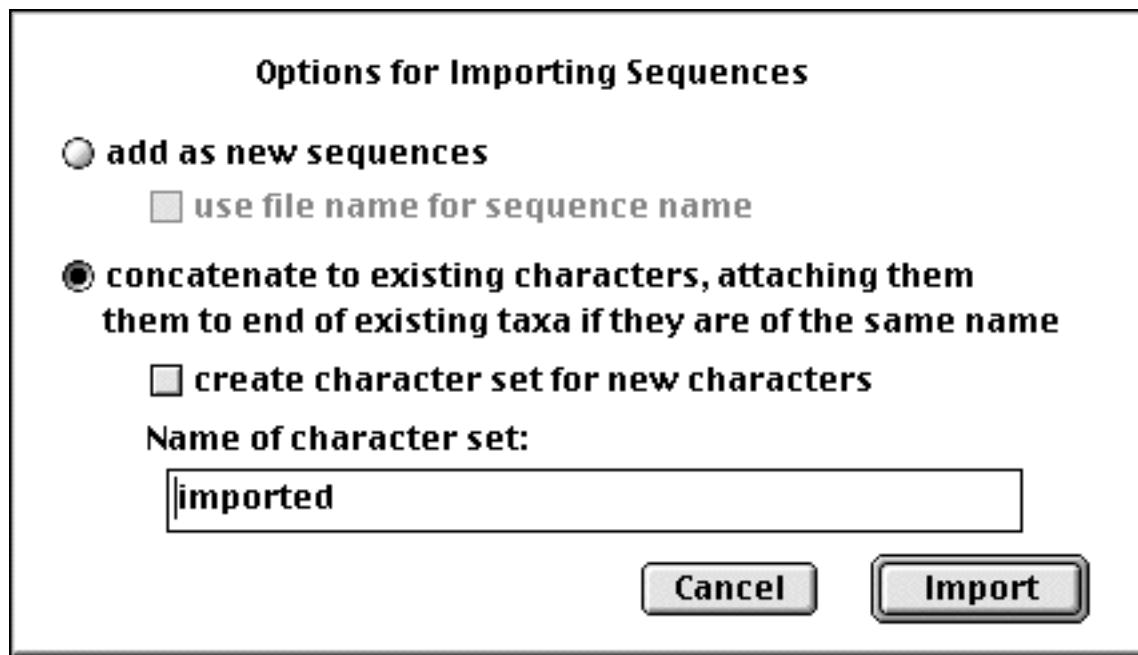
Importing sequences into an existing file

If you have a molecular sequence data file already open, and you are in MacClade's data editor, then MacClade can import additional sequences into that file using the **Import Sequences** submenu of the **Taxa** menu. Sequences to be imported to should be in either GenBank, NBRF, FASTA, File with Name & Sequence, or File with Sequence Only formats. The first three formats are standard, and if those are used multiple sequences can be present in the file to be imported. The last two are simple formats which might prove useful if you need to prepare the file yourself, but they can contain information for only a single sequence.

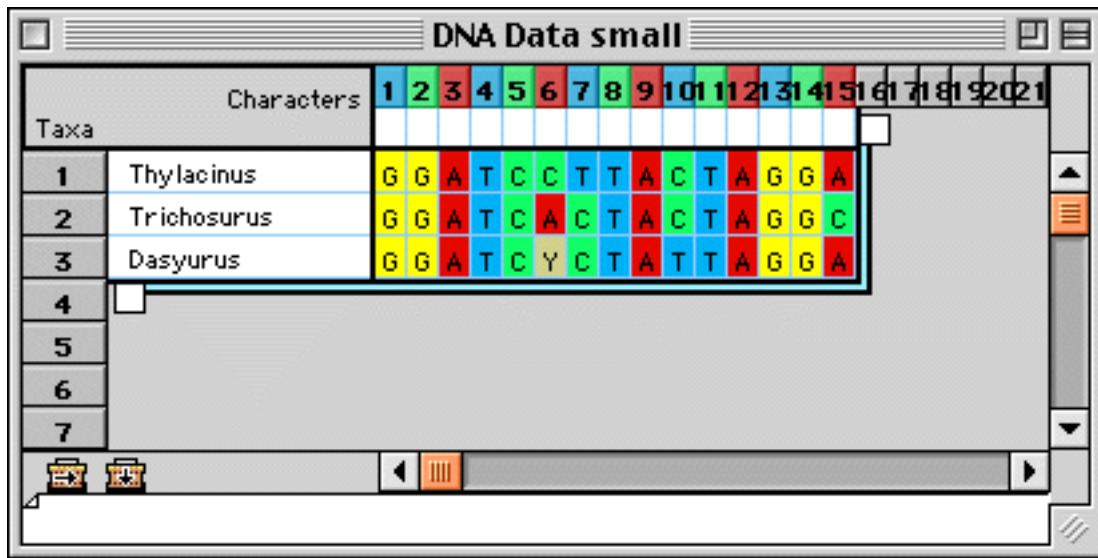
For the File with Sequence Only option, the file can consist of nothing other than raw sequence data. MacClade will use the file name as the taxon name.

For the File with Name & Sequence option, the first token (word) in the file should be the taxon name; the remainder of the file can consist of nothing but the sequence for that taxon; spaces, tabs, and new line characters are ignored.

If you import files in GenBank, NBRF, FASTA, or File with Name & Sequence format, you will be presented with a dialog box querying exactly how the imported sequence(s) are to be treated:



You can choose either to add the newly imported data as entirely new sequences ("add as new sequences") or to add them on to the end of an existing taxon if there is a taxon with a matching name. For example, if you begin with the following matrix:

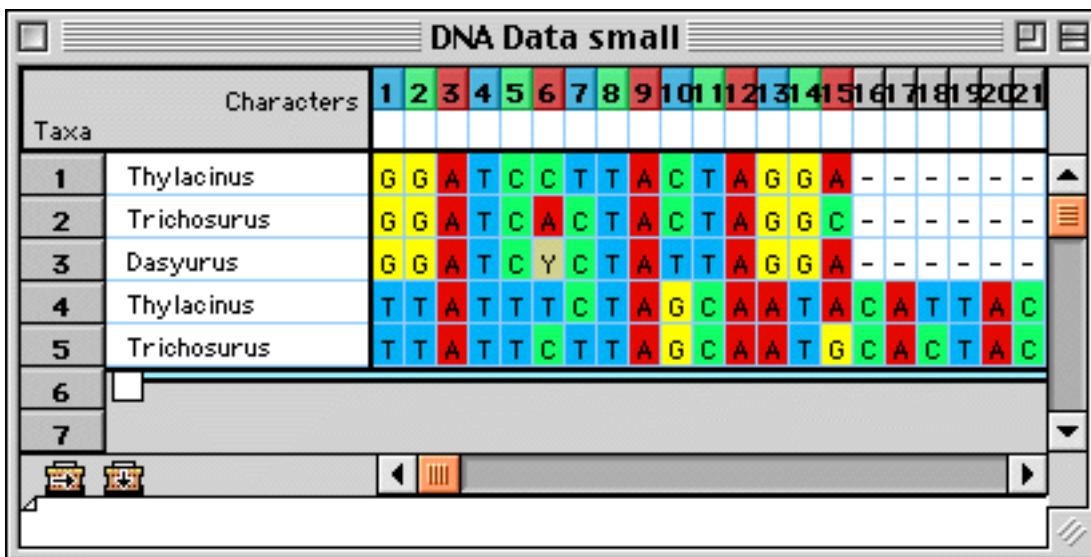


and ask to import the following NBRF file with two sequences

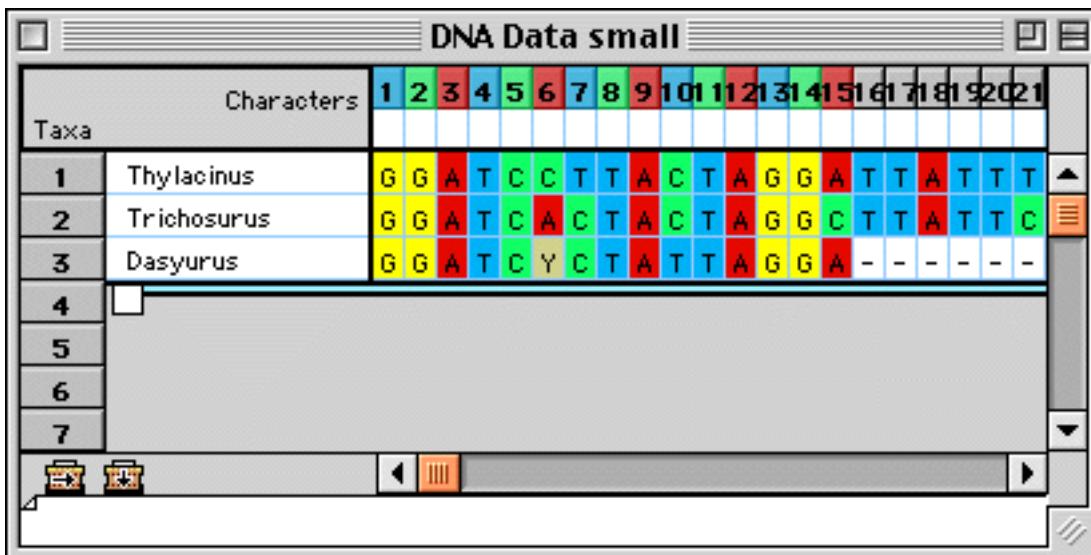
```
>DL; Thylacinus
Thylacinus
TTATTCTAGCAATACATTACACATCA
GACACATCAACTGCCTCTCCTCAGTA
GCACATATCTGTC*
```

```
>DL; Trichosurus
Trichosurus
TTATTCTTAGCAATGCACACTACACCGCT
GATACAGCAACAGCTTCTCATCAGTA
GCCACATCTGTC*
```

and ask to have these data added as new sequences, then two new sequences would be added to the matrix:



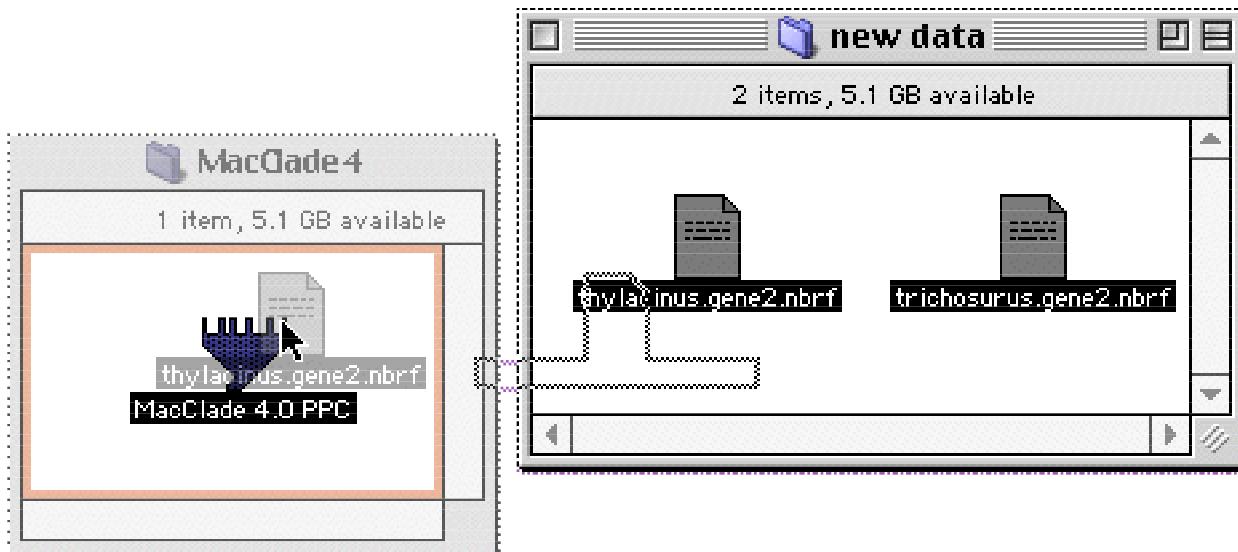
If instead you asked to import the sequences and concatenate them on to the end of existing characters, MacClade would discover that there already were taxa in the matrix named Thylacinus and Trichosurus, and it would concatenate the new data onto the end of those taxa:



In this latter case, you also have the option of having MacClade create a character set (see "[Defining your own character sets](#)" on page 201) for the newly created characters, so that you may more easily manipulate them in the future.

Importing sequences from multiple files at once

If you have a data file with molecular sequences open in MacClade, you can have multiple sequences from different files imported all together if you select those files in the Finder and drop them onto the MacClade application icon:



MacClade will query about the format of the sequences to be imported. Only files of the formats GenBank, NBRF, FASTA, File with Name & Sequence, or File with Sequence Only are allowed. All files to be imported at one time must be of the same format.

By default, MacClade will not allow this if the data files to be imported have MacClade icons (more formally, they have their creators set to 'CLAD'). If you wish to import NBRF, GenBank, FASTA, Name & Sequence, or Sequence Only files that have been set to have MacClade icons, you will need to choose **Options** in the **Import Sequences** submenu and set the option to "allow importation from Finder of sequences from files of type 'CLAD'". This will *not* allow you to import NEXUS files in this fashion, only files of those other formats.

Exporting descriptions

If you ask MacClade to export a file in Descriptions format, it will write out the information in the data matrix in telescopic prose. For example, the Descriptions format for the "Amblygnathus" file included in the Examples folder begins something like this:

```
seriatoporus-g
m.l. vent. flange present. elyt. micro. isodiametric. body length 6.3-7.0mm. pron. hind ang. rounded. # int sac
spines numerous. spine size large. spine loc'n scattered. prostern. process not. head micros. isodium. elytral vest.
8-9. head & pron color black. clypeal margin straight. left mand. apex normal. spermatheca short curve. male
lamina width broad. lamina length moderate. lam. apex dir. right. apex shape pointed. m.l. apex form mod
straight. m.l. flange bilo not. dorsal flange small. female terg.8 setae absent. sac microtrich. normal. leg color fla-
vous. m.l. vent.-sculp. smooth. elytral color black. m.l. vent.-form tapered. pron. base moderate.
```

ellipticus-g
m.l. vent. flange present. elyt. micro. isodiametric. body length 4.5-6.2mm. pron. hind ang. subangulate. # int sac spines none. spine size absent. spine loc'n absent. prostern. process ridged. head micros. isodiam. elytral vest. 9 & ap1-8. head & pron color black. clypeal margin straight. left mand. apex normal. spermatheca short curve. male lamina width broad. lamina length long. lam. apex dir. right. apex shape pointed. m.l. apex form mod straight. m.l. flange bilo not. dorsal flange small. female terg.8 setae absent. sac microtrich. normal. leg color flavous. m.l. vent.-sculp. smooth. elytral color black. m.l. vent.-form tapered. pron. base moderate.

MacClade cannot read files in Descriptions format.

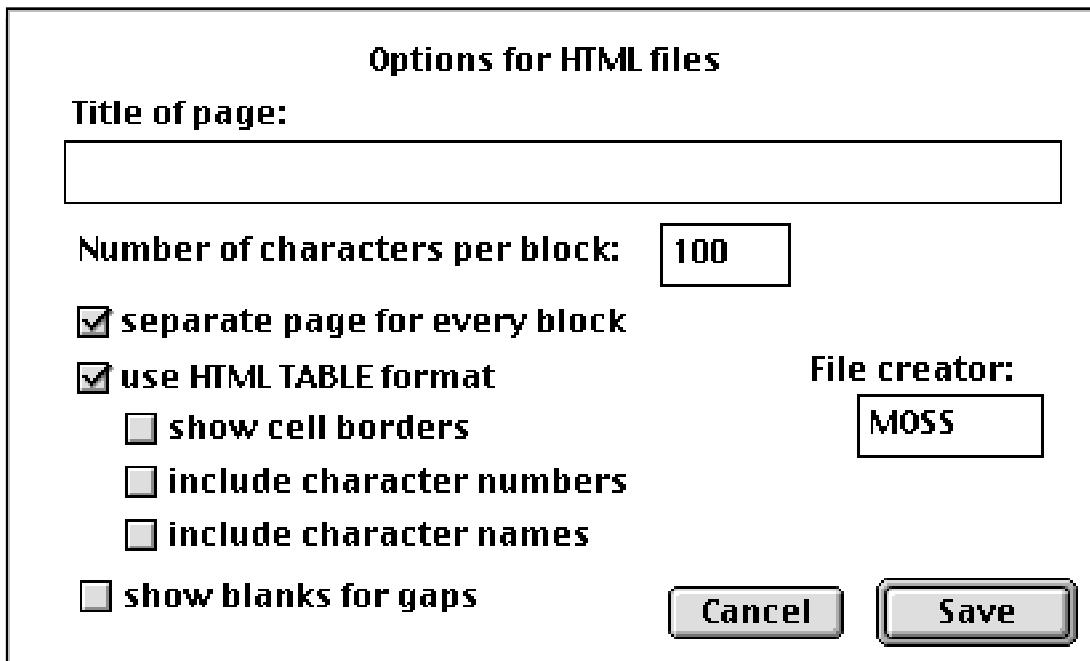
Exporting MEGA files

Molecular sequence data can be exported to the format used by MEGA version 1 (Kumar et al., 1993), using the **MEGA 1** item in the **Export File** submenu in the **File** menu. If some terminal taxa are polymorphic or have partial uncertainty at some sites, then MEGA 1 will not be able to read the files without modification. If MacClade detects polymorphisms or partial uncertainties, it will provide three options: to cancel the export, to convert polymorphisms and uncertainties to missing data, or to write the polymorphisms and uncertainties using IUPAC ambiguity codes. The latter option will produce files that are incompatible with MEGA version 1, but which will be compatible with a later version (Kumar, pers. comm.).

Exporting Web pages (HTML files)

The **HTML File** item of the **Export File** submenu allows you to export data matrices in HyperText Markup Language (HTML), the format used on typical World Wide Web pages. These files can then be moved onto a Web server to make them generally available.

There are several options for the format of the HTML files produced, which can be set in the dialog box that appears when you choose the HTML File export command:



The title of the HTML page should be entered into the "Title of page" box. For the "Primate mtDNA" example file, if you enter "Primate mtDNA" as the title of the page, the default HTML file will look as follows in a Web browser such as Netscape Navigator:

[First] [Previous] [Next]

[1-100] [101-200] [201-300] [301-400] [401-500] [501-600] [601-700] [701-

Primate mtDNA

Sites 1 through 100

| | * | * | * |
|-------------------------|--------------------------------------|---|---|
| Homo sapiens | AAGCTTCACCGGGCGCAGTCATTCTCATTAATCGCC | | |
| Pan | AAGCTTCACCGGGCGCAATTATCCTCATTAATCGCC | | |
| Gorilla | AAGCTTCACCGGGCGCAGTTGTTCTTATAATTGCC | | |
| Pongo | AAGCTTCACCGGGCGCAACCAACCTCATGATTGCC | | |
| Hylobates | AAGCTTTACAGGTGCAACCGTCCTCATTAATCGCC | | |
| Macaca fuscata | AAGCTTTCCGGCGCAACCAATCCTTATGATCGCT | | |
| M. mulatta | AAGCTTTCTGGCGCAACCAATCCTCATGATTGCT | | |
| M. fascicularis | AAGCTTCTCCGGCGCAACCAACCTTATAATCGCC | | |
| M. sylvanus | AAGCTTCTCCGGTGCAACTATCCTTATAGTTGCC | | |
| Saimiri sciureus | AAGCTTCACCGGGCGCAATGATCCTAATAATCGCT | | |
| Tarsius syrichta | AAGTTTCATTGGAGCCACCTCTTATAATTGCC | | |
| Lemur catta | AAGCTTCATAGGAGCAACCAATTCTAATAATCGCA | | |

MacClade writes the character data in blocks; by default, the size of a block is 100 characters. If "separate page for each block" is checked, each block will be written on a separate HTML page, with links between the pages. If "separate page for each block" is unchecked, all of the data will be written on one page.

If "use HTML table format" is turned off, the HTML file is written without using HTML tables; instead, <PRE> </PRE> tags are used, with colored text:

Primate mtDNA

Sites 1 through 100

| | * | * | * | |
|-------------------------|---|---|---|--|
| Homo sapiens | AAGCTTCACCGGGCGCAGTCATTCTCATTAATCGCCCCA | | | |
| Pan | AAGCTTCACCGGGCGCAATTATCCTCATTAATCGCCCCA | | | |
| Gorilla | AAGCTTCACCGGGCGCAGTTTCTTCTTATAAATTGCCCA | | | |
| Pongo | AAGCTTCACCGGGCGCAACCACCCCTCATGATTGCCCA | | | |
| Hylobates | AAGCTTTACAGGTGCAACCCTCCTCATTAATCGCCCCA | | | |
| Macaca fuscata | AAGCTTTCCGGCGCAACCATCCTTATGATTGCTCA | | | |
| M. mulatta | AAGCTTTCTGGCGCAACCATCCTCATGATTGCCCA | | | |
| M. fascicularis | AAGCTTCTCCGGCGCAACCACCCCTTATAAATTGCCCA | | | |
| M. sylvanus | AAGCTTCTCCGGTGCACACTATCCTTATAAGTTGCCCA | | | |
| Saimiri sciureus | AAGCTTCACCGGGCGCAATGATCCTTATAAATTGCCCA | | | |
| Tarsius syrichta | AAGTTTCATTGGAGCCACCACTCTTATAAATTGCCCA | | | |
| Lemur catta | AAGCTTCATAGGAGCAACCAATTCTAATAATCGCACCA | | | |

The appearance of the matrix in this web page is also somewhat sensitive to the changes you make in the display of the data in MacClade's data editor. For example, if you display protein-coding DNA sequence data with the color of the cells representing the translated amino acid, those colors will be used in the HTML file:

Primate mtDNA

Sites 1 through 100

| | * | * | * | |
|-------------------------|--|---|---|--|
| Homo sapiens | AAGCTTCACCGGGCGCAGTCATTCTCATTAATCGCCG | | | |
| Pan | AAGCTTCACCGGGCGCAATTATCCTCATTAATCGCCG | | | |
| Gorilla | AAGCTTCACCGGGCGCAGTTTCTTCTTATAAATTGCCG | | | |
| Pongo | AAGCTTCACCGGGCGCAACCACCCCTCATGATTGCCG | | | |
| Hylobates | AAGCTTTACAGGTGCAACCCTCCTCATTAATCGCCG | | | |
| Macaca fuscata | AAGCTTTCTGGCGCAACCATCCTTATGATCGCCG | | | |
| M. mulatta | AAGCTTTCTGGCGCAACCATCCTCATGATTGCCG | | | |
| M. fascicularis | AAGCTTCTCCGGCGCAACCACCCCTTATAAATCGCCG | | | |
| M. sylvanus | AAGCTTCTCCGGTGCACACTATCCTTATAAGTGGCG | | | |
| Saimiri sciureus | AAGCTTCACCGGGCGCAATGATCCTTATAAATCGCCG | | | |
| Tarsius syrichta | AAGTTTCATTGGAGCCACCACTCTTATAAATTGCCG | | | |
| Lemur catta | AAGCTTCATAGGAGCAACCAATTCTAATAATCGCACG | | | |

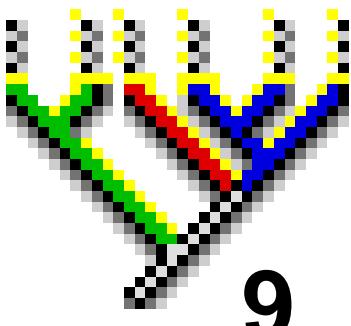
If states names are shown in the editor, and two options in the HTML file export dialog box are chosen ("show cell borders" and "include character names"), a morphological matrix might yield an HTML file that is displayed as follows:

Amblygnathus Morphological Matrix

Characters 1 through 28

| | m.l. vent. flange | elyt. micro. | body length | pron. hind ang. | # int sac spines |
|-----------------------|-------------------------|-----------------|----------------|--------------------|------------------------|
| seriatoporus-g | present | isodiametric | 6.3-7.0mm | rounded | numerous |
| ellipticus-g | present | isodiametric | 4.5-6.2mm | subangulate | none |
| amaroides-g | present | transverse | 4.5-6.2mm | subangulate | none |
| ruficollis-g | absent | transverse | 4.5-6.2mm | angulate | numerous |
| iripennis | absent | transverse | 4.5-6.2mm | subangulate | one |

If "show blanks for gaps" is chosen, then any gap is shown as a blank space; otherwise, it is shown as the gap symbol.



MACCLADE'S WINDOWS

9

MacClade has several windows in which you can edit data, examine and manipulate trees, study character evolution, and so on. This chapter provides an overview of the windows, with links to the more detailed information found elsewhere.

The Windows menu

There are two dominant windows in MacClade, the data editor and the tree window. One of these two windows is usually open when you are working on a MacClade file; they can't both be open at the same time. The first two items in the **Windows** menu in MacClade allows you to switch between the data editor and the tree window.

In addition to switching between the data editor and tree window, the **Windows** menu allows you to open two windows for editing textual notes about the data file and trees, as well as open or close the tool palettes associated with the data editor and tree window. The menu also lists any other open windows; if you choose their menu items, they will be moved to the front of the remaining windows.

Closing windows

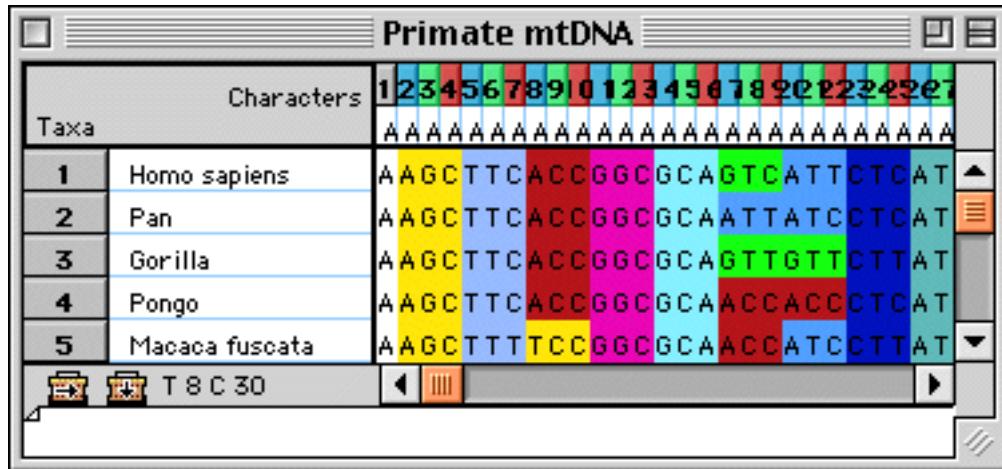
Windows are closed either by touching their go-away box (the small box in the upper left corner) or by choosing **Close** from the **File** menu. Note that **Close** will not close the file unless you use it on the last open window displayed for the file.

Data editor

MacClade's data editor provides a matrix of cells in which you can enter and edit your data. There are many different display modes, some designed for nonmolecular data:

| Characters | | 1 | 2 | 3 |
|------------|----------------|-------------------|--------------|-------------|
| Taxa | | m.l. vent. flange | elyt. micro. | body length |
| 1 | seriatoporus-g | present | isodiametric | 6.3-7.0mm |
| 2 | ellipticus-g | present | isodiametric | 4.5-6.2mm |
| 3 | amaroides-g | present | transverse | 4.5-6.2mm |
| 4 | ruficollis-g | absent | transverse | 4.5-6.2mm |
| 5 | iripennis | absent | transverse | 4.5-6.2mm |
| 6 | mexicanus | absent | transverse | 4.5-6.2mm |

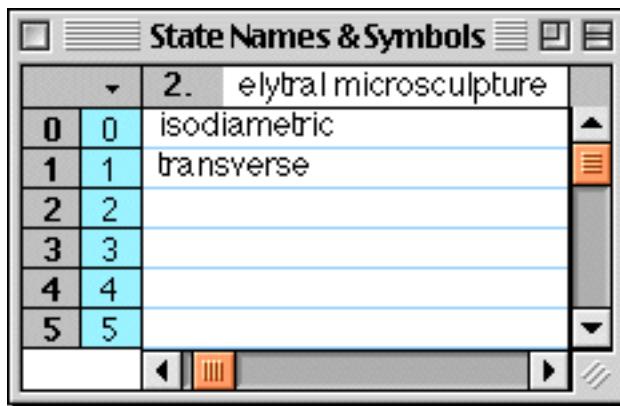
and others specifically designed for molecular sequences:



In the editor you can move rows and columns, enter data in cells, recode data, search and replace, conduct manual sequence alignment, and so on. More details can be found in the chapters on ["Entering and Editing Data"](#) and ["Preparing Molecular Data"](#).

State names & symbols window

The names of characters and their states are edited in the State Names & Symbols window, available from the **Characters** menu:



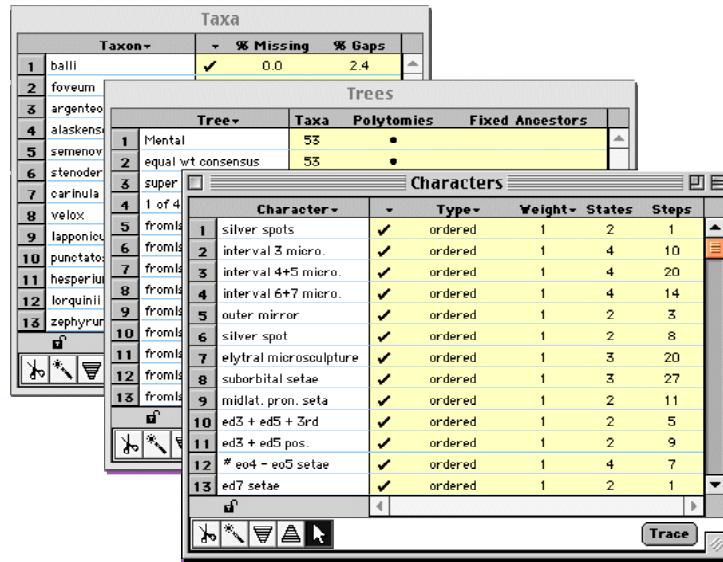
In this window you can scroll from one character to the next, and edit their names. This window is not available for molecular sequence data; for those data you will need to use the data editor to name characters should you wish.

This window also allows you to edit the symbols used to represent the various states.

More details are given in [Chapter 12](#).

List windows

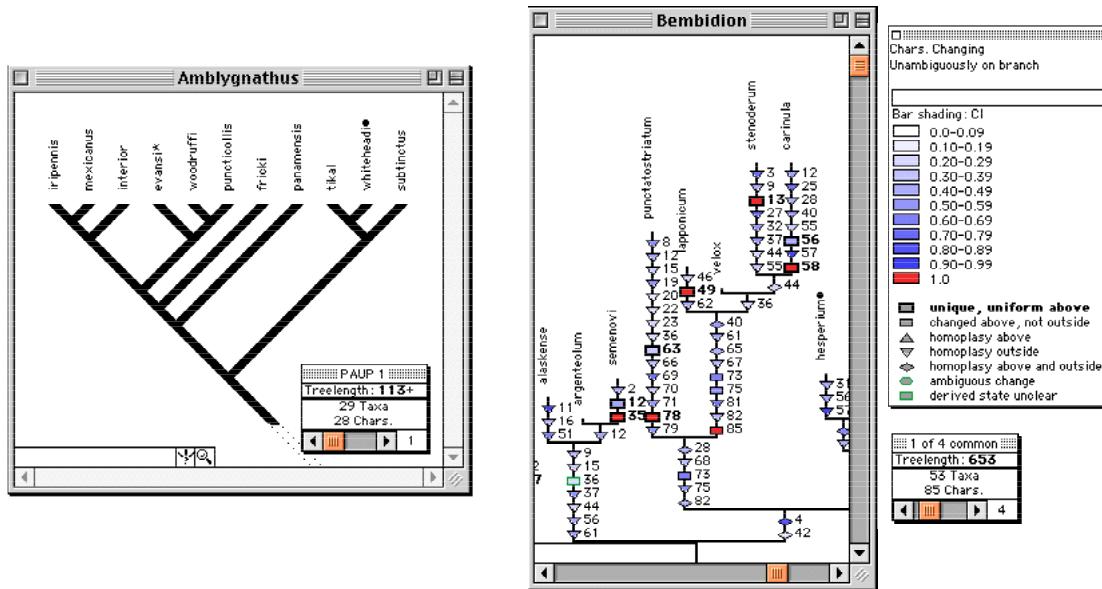
Several windows showing lists of objects (characters, taxa, trees, and so forth) are available in MacClade to edit or observe properties of those objects:



These list windows all share a common set of tools and behaviors, discussed in [Chapter 10](#). All of the list windows allow you to rearrange or delete the objects, edit their names, and see various properties of the objects. The nine list windows in MacClade list information about characters, character sets, taxa, taxon sets, trees, weight sets, type sets, inclusion sets, and NEXUS blocks. See [Chapter 10](#) and the chapters cited therein for more information.

Tree window

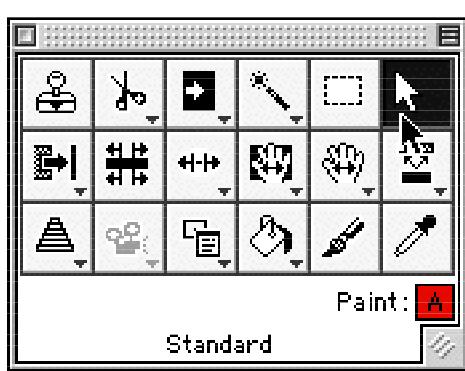
The tree window is the main analysis window in MacClade. This is the window in which you can move branches on trees, reroot them, trace character evolution on them, and so on.



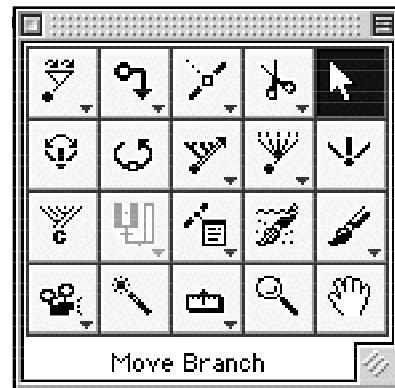
[Chapters 17](#) through [23](#) provide details about the tree window.

Tool palettes

Both the data editor and tree window have tool palettes associated with them. These tool palettes contain tools to manipulate or examine data or trees and characters:



Tool palette for data editor



Tool palette for tree window

Details about each of these tools can be found in the chapters about the data editor ([Chapter 13](#) and [Chapter 16](#)) and the tree window ([Chapter 17](#), [Chapter 18](#), and [Chapter 24](#)).

Tool palettes can be closed by clicking on their go-away box, or by choosing **Tool Palette** from the **Windows** menu. If a tool palette has been closed, and thus is no longer on the screen, it can be reopened by choosing **Tool Palette** from the **Windows** menu.

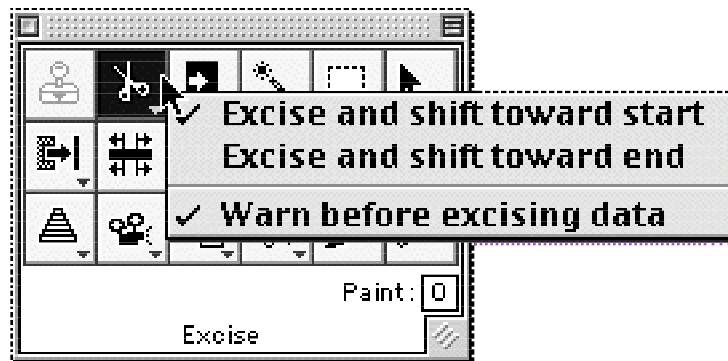
The tool palettes can also be resized. If you wish, you may resize them so that they are horizontal rulers across the bottom of the screen:



or are vertical rulers against the left edge, or form some other arrangement.

To choose a tool, select it by touching it in the tool palette. At the bottom of the tool palette will be a short textual description of the tool's action.

Some tools have various options which alter how they work. The options are chosen using a little pop-up menu that appears if you touch on the tool and hold the mouse button down:



All tools with a little triangle in the lower right of their square:

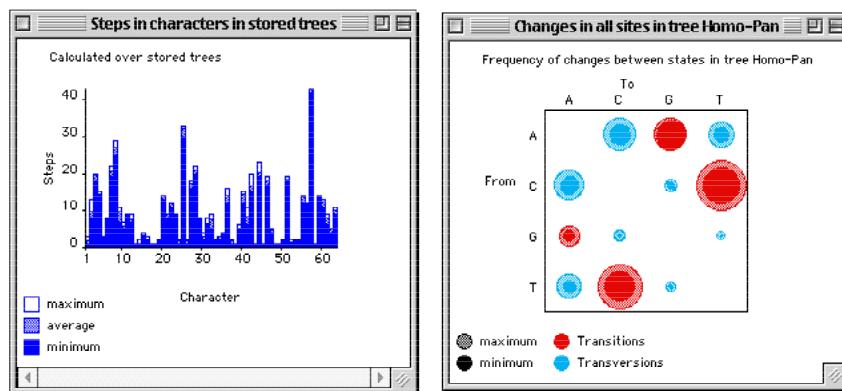


have a pop-up menu in which various options can be chosen.

Many tools have two primary modes, and you can toggle between these modes using the pop-up menu or by holding down the Option key. Alternative primary modes have slightly different cursors.

Chart window

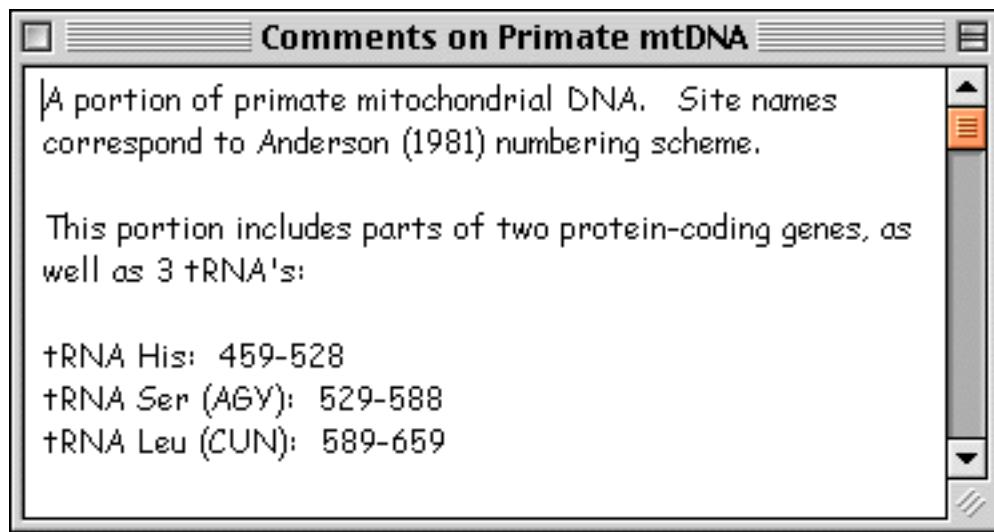
Calculations conducted using the items in the **Charts** menu are presented in a chart window. These calculations summarize aspects of characters or character evolution across multiple characters or multiple trees.



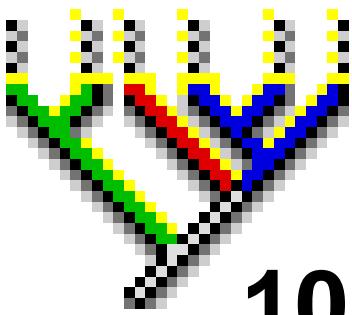
Charts are available only if the tree window is open. [Chapter 20](#) gives full details about charts and the chart window.

Text windows for keeping notes

Two text windows, available from the **Windows** menu, are provided so that you might enter notes about the data file or tree file.



These windows are discussed further in [Chapter 24](#).



LIST WINDOWS

10

MacClade deals with objects of different kinds, including taxa, characters, and trees. There is a common interface, the list window, for manipulating such objects.

The list windows

There are nine list windows in MacClade:

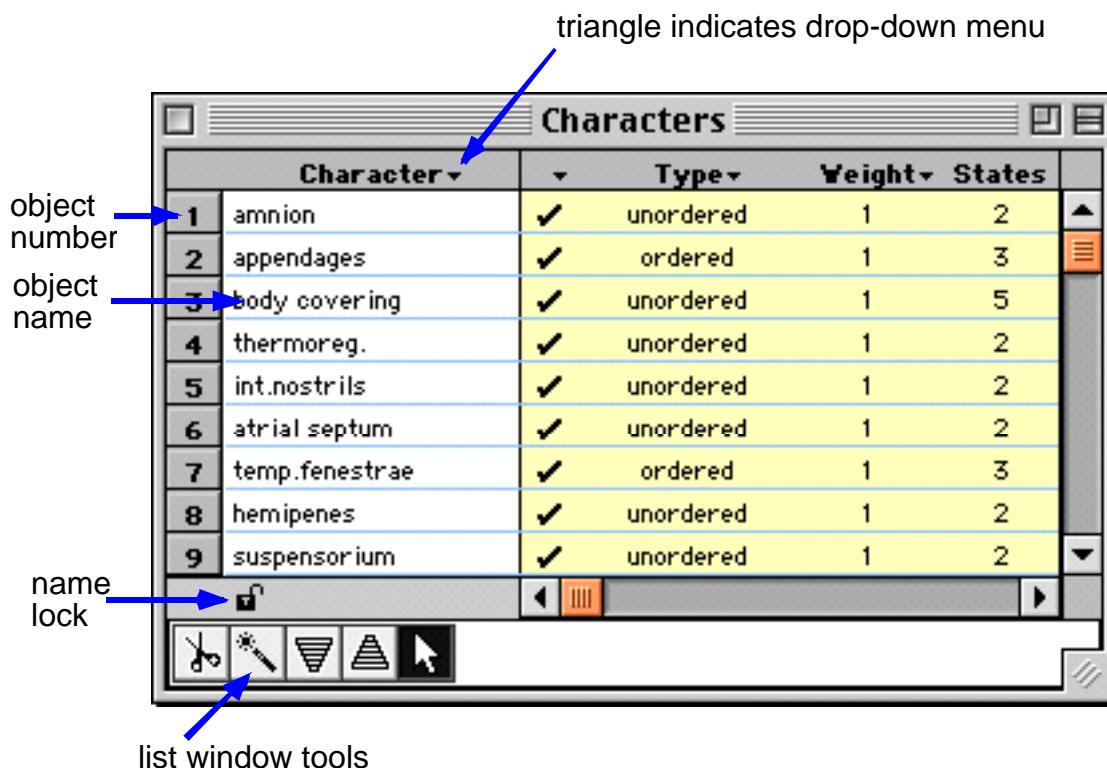
1. The **Character List** window, available from the **Characters** menu, which lists characters. This window allows you to include or exclude characters from analyses, set their weights, observe their number of steps, and so on, as discussed in more detail in [Chapter 12](#) and [Chapter 15](#).
2. The **Character Sets List** window, available from the **Character Sets** submenu of the **Characters** menu, which lists character sets. Character sets allow you to select characters quickly in the character list window, and are useful if you use MacClade in conjunction with PAUP* (see [Chapter 12](#)).
3. The **Weight Sets List** window, available from the **Weight Sets** submenu of the **Characters** menu. This window lists weight sets you have stored (see [Chapter 15](#)).
4. The **Type Sets List** window, available from the **Type Sets** submenu of the **Characters** menu. This window lists type sets you have stored (see [Chapter 15](#)).
5. The **Inclusion Sets List** window, available from the **Inclusion Sets** submenu of the **Characters** menu. This window lists character inclusion sets you have stored (see [Chapter 15](#)).
6. The **Taxon List** window, available from the **Taxa** menu, allows you to include or exclude taxa from trees, and see various properties of taxa (see [Chapter 11](#)).
7. The **Taxon Sets List** window, available from the **Taxon Sets** submenu of the **Taxa** menu, which lists taxon set. Taxon sets allow you to select taxa quickly in the taxon list window, and are useful if you use MacClade in conjunction with PAUP* (see [Chapter 11](#)).
8. The **Tree List** window, available from the **Trees** menu, which lists stored trees (see [Chapter 17](#)).
9. The **NEXUS Block List** window, available from the **NEXUS Blocks** submenu of the **Edit** menu, which lists NEXUS blocks contained in a file (see [Chapter 7](#)).

With a few exceptions, the behavior of each of these list windows and the tools and commands available to manipulate the lists is constant from window to window. The most notable exception is the NEXUS block list window, which is much more limited in available editing tools.

This chapter describes the common elements of each of these windows, and a few of the features unique to the character and taxon list windows. A few other features are described in the chapters mentioned in the above list.

Structure of a list window

A list window displays a list of objects, as shown in the figure below for the character list window.



On the left edge the number of each object is displayed, followed on the right by the name of the object, and then by various columns indicating properties of the object, as described in the title for the column. The object number and name are displayed in all list windows; the remaining columns vary in content depending upon the objects being listed.

Some of the titles for the columns have small triangles, indicating that a drop-down menu is available and will appear if you touch on the title name. Use of these menus will be described in later sections.

List window tools

List windows have a tool bar at their bottom:



These tools act upon the rows or columns of the list window. Make sure, if you wish to use one of these tools on a list window's rows, you choose the tool from the bottom left of that list window, not from another list window (each list window has its own active tool), and not from the data editor tool palette or the tree window tool palette.

The tools are:

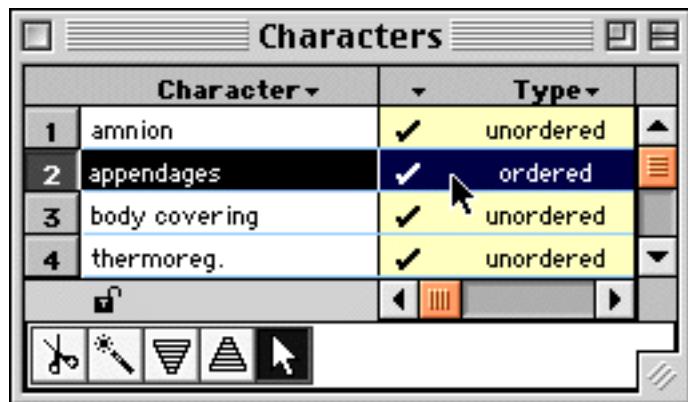
| Tool | Function |
|------|--|
| | The scissors tool permanently deletes objects. Described in " Deleting objects " on page 178. |
| | The Selection Wand is used to select objects in the list window based on their values in a column, Described in " Selecting similar objects " on page 170. |
| | The sort descending tool will sort the objects in the list window based on their values in a column, with largest values at the top and smallest values at the bottom. Described in " Sorting objects automatically " on page 177. |
| | The sort ascending tool will sort the objects in the list window based on their values in a column, with smallest values at the top and largest values at the bottom. Described in " Sorting objects automatically " on page 177. |
| | The standard arrow tool. |

The functions of these tools are discussed throughout the remainder of this chapter.

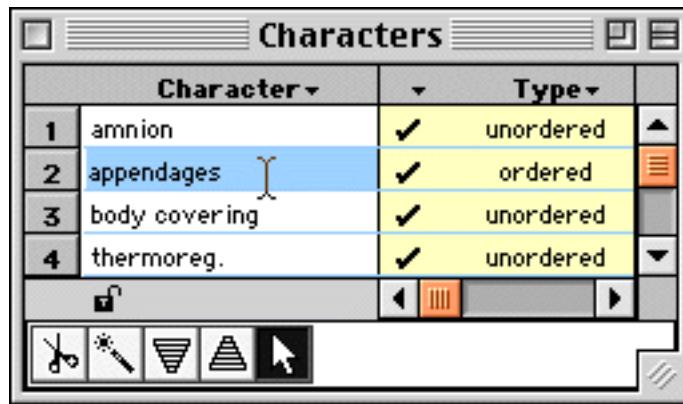
Selecting objects

An important aspect of MacClade 4 is its ability to select objects in list windows, as it allows quick manipulation of taxa, characters, trees, and so on. A simpler version of this ability was present (and, for the most part, unnoticed) in MacClade 3's Character Status window. MacClade 4, however, extends this capability significantly.

When we speak of selecting an object in a list window, we mean selecting the entire row in the list window:

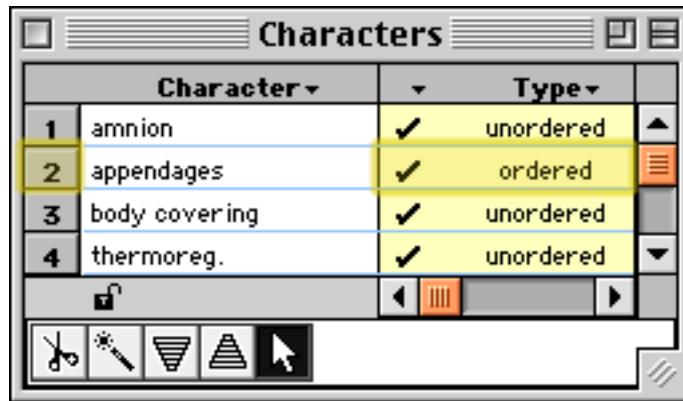


as opposed to just selecting the objects name so that the name can be edited:



Manually selecting rows

To select a row in a list window, touch on any part of the row *other than the name* of the object with the arrow tool. For example, to select the second row in the following example, touch on either the row number or the yellow portion of the row shown to the right of the name, as shown below.



Touching on the regions bordered in yellow will select the second row

By default, touching on the name of the object will cause just the name to be selected for renaming. If the names are locked ([page 179](#)), however, touching anywhere on the row will cause it to be selected.

If a row is already selected (for example, the taxon 2 in the example below), and you click on another row (for example, taxon 5), then taxon 2 will be de-selected, and taxon 5 selected:

| Taxon | % Missing |
|---------------|-----------|
| 1 balli | 0.0 |
| 2 foveum | 3.5 |
| 3 argenteolum | 25.9 |
| 4 alaskense | 0.0 |
| 5 semenovi | 38.8 |
| 6 stenoderum | 31.8 |

click

| Taxon | % Missing |
|---------------|-----------|
| 1 balli | 0.0 |
| 2 foveum | 3.5 |
| 3 argenteolum | 25.9 |
| 4 alaskense | 0.0 |
| 5 semenovi | 38.8 |
| 6 stenoderum | 31.8 |

If instead you hold down the Shift key as you click on taxon 5, the selection will be extended, in the standard Macintosh way, to include all of the taxa from 2 through 5:

| Taxon | % Missing |
|---------------|-----------|
| 1 balli | 0.0 |
| 2 foveum | 3.5 |
| 3 argenteolum | 25.9 |
| 4 alaskense | 0.0 |
| 5 semenovi | 38.8 |
| 6 stenoderum | 31.8 |

Shift-click

| Taxon | % Missing |
|---------------|-----------|
| 1 balli | 0.0 |
| 2 foveum | 3.5 |
| 3 argenteolum | 25.9 |
| 4 alaskense | 0.0 |
| 5 semenovi | 38.8 |
| 6 stenoderum | 31.8 |

Finally, if you hold down the ⌘ key, the row you touch upon will be selected if it was previously unselected, and de-selected if it was selected, without affecting the selection of any other rows:

| Taxon | % Missing |
|---------------|-----------|
| 1 balli | 0.0 |
| 2 foveum | 3.5 |
| 3 argenteolum | 25.9 |
| 4 alaskense | 0.0 |
| 5 semenovi | 38.8 |
| 6 stenoderum | 31.8 |

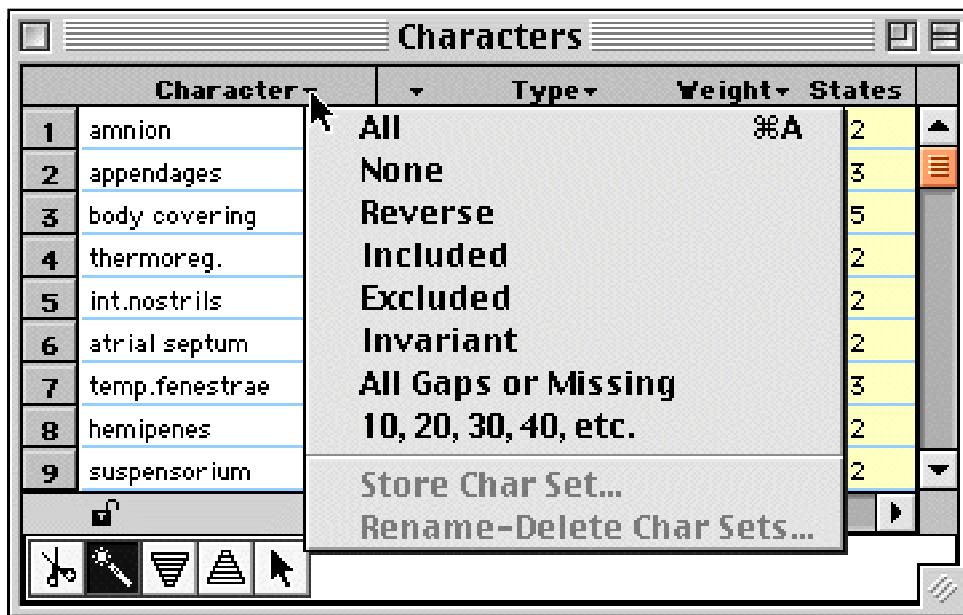
 ⌘ click

| Taxon | % Missing |
|---------------|-----------|
| 1 balli | 0.0 |
| 2 foveum | 3.5 |
| 3 argenteolum | 25.9 |
| 4 alaskense | 0.0 |
| 5 semenovi | 38.8 |
| 6 stenoderum | 31.8 |

Selecting or de-selecting sets of objects

You can quickly select or de-select sets of objects using the **Select** submenu. It is available whenever a list window is frontmost. For the character list and taxon list, these sets include both predefined sets and those character sets and taxon sets, respectively, that you define yourself.

The **Select** submenu is available in two places: (1) as a submenu in the **Edit** menu; (2) as a menu that drops down when you touch on the title of the object in the list window (the title "Character" or "Site" for the character list window, "Taxon" for the taxon list window, "Tree" for the tree list window, and so on), as shown in the example below:



Three commands are present in the **Select** submenu of all list windows: **All**, **None**, **Reverse**. If **All** is chosen, the entire list is selected. If **None** is selected, the entire list is de-selected. If **Reverse** is chosen, those items already selected will become unselected, and vice versa. For the character list and taxon list, there are additional commands available in the **Select** submenu.

Predefined character sets

For the character list, the additional, predefined sets of characters are:

Included: If this is chosen, all included characters will be selected.

Excluded: This is the set of all excluded characters.

Invariant: If this is chosen, then all characters containing at most one state will be selected. A character is considered invariant by MacClade if it has only one state. Missing data and gaps are not counted as states in this definition, and thus the character will be considered invariant if some taxa have state 0, others have missing data, and the remaining have gaps.

All Gaps or Missing: If this is chosen, all characters containing only gaps or missing will be selected.

10, 20, 30, 40, etc.: If this is chosen, every tenth character will be selected. This character set is not intended for use in this context, as it seems unlikely selecting every tenth character in the list win-

dow would be useful; it is intended for the character set shading feature in the data editor (["Shading character sets" on page 266](#)).

Gap Runs... (available only for molecular sequence data): If you choose this, you will be presented with a dialog box, which will allow you to select in the character list window all characters that participate in runs of gaps of defined lengths. This does not include terminal runs of gaps, at either end of a sequence. For example, in the following matrix, if you asked to select characters participating in gap runs of length 2 or more, you would select characters 3-5, 10-14, 18-19, as shown in the row marked "gap run of 2". If you increase the length of the gap runs desired, fewer and fewer characters would be selected.

| | | | | | | | | | | | | | | | | | | | | |
|----------|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| taxon 1 | a | c | g | c | A | C | G | A | T | - | - | - | - | A | - | A | - | - | A | |
| taxon 2 | - | C | - | C | G | C | G | A | T | T | T | G | C | A | G | A | C | A | - | A |
| taxon 3 | - | C | C | C | t | c | g | a | t | - | - | - | - | A | - | A | - | - | A | |
| taxon 4 | - | C | - | C | - | C | G | A | t | - | - | - | - | A | - | A | - | - | A | |
| taxon 5 | - | - | - | C | c | g | a | t | T | - | - | - | a | - | a | - | a | - | A | |
| taxon 6 | - | - | - | g | c | A | C | g | A | T | - | - | - | - | A | - | A | - | A | |
| taxon 7 | - | - | - | g | c | A | C | g | A | T | - | - | - | - | A | G | A | - | A | |
| taxon 8 | - | - | - | C | - | C | G | A | T | - | - | - | - | A | - | A | - | - | A | |
| taxon 9 | - | c | - | - | C | G | A | T | T | T | - | T | A | C | A | - | A | - | A | |
| taxon 10 | - | - | - | - | C | G | A | T | - | - | - | - | A | - | A | - | - | A | | |

gap run of 2:   
 gap run of 3:  
 gap run of 4 or 5: 
 gap run of 6 or more: (none selected)

In general, if you ask for gap runs of length x, then if there is, in any taxon, a string of contiguous gaps of length greater than or equal to x running from character y to character z, then characters y through z would be selected. Again, this does not apply if y is the first character or z the last.

Protein Coding (available only with DNA or RNA sequence data): This will select all bases designated as protein-coding.

Non-Coding (available only with DNA or RNA sequence data): If chosen, characters that you have not designated to be protein-coding will be selected.

Every 1st Position (available only with DNA or RNA sequence data): This will select all positions designated as first positions of codons.

Every 2nd Position (available only with DNA or RNA sequence data): This will select all positions designated as second positions of codons.

Every 3rd Position (available only with DNA or RNA sequence data): This will select all positions designated as third positions of codons.

Options for defining coding and non-coding regions, and codon positions, are discussed in [Chapter 16](#).

Predefined taxon sets

For the taxon list, additional predefined sets of characters include:

Included in tree (available only while the tree window is open): This will cause all taxa in the current tree to be selected.

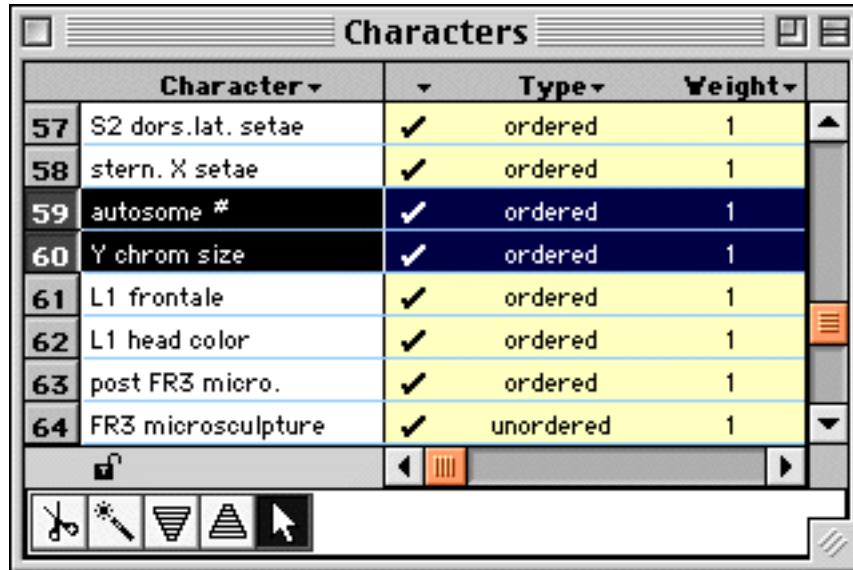
Excluded from tree (available only while the tree window is open): This will cause all taxa *not* in the current tree to be selected.

For characters and taxa you may also define your own character sets and taxon sets. These can be very powerful for quick manipulation of characters and taxa; for full details, see "[Defining your own character sets](#)" on page 201 and "[Defining your own taxon sets](#)" on page 189.

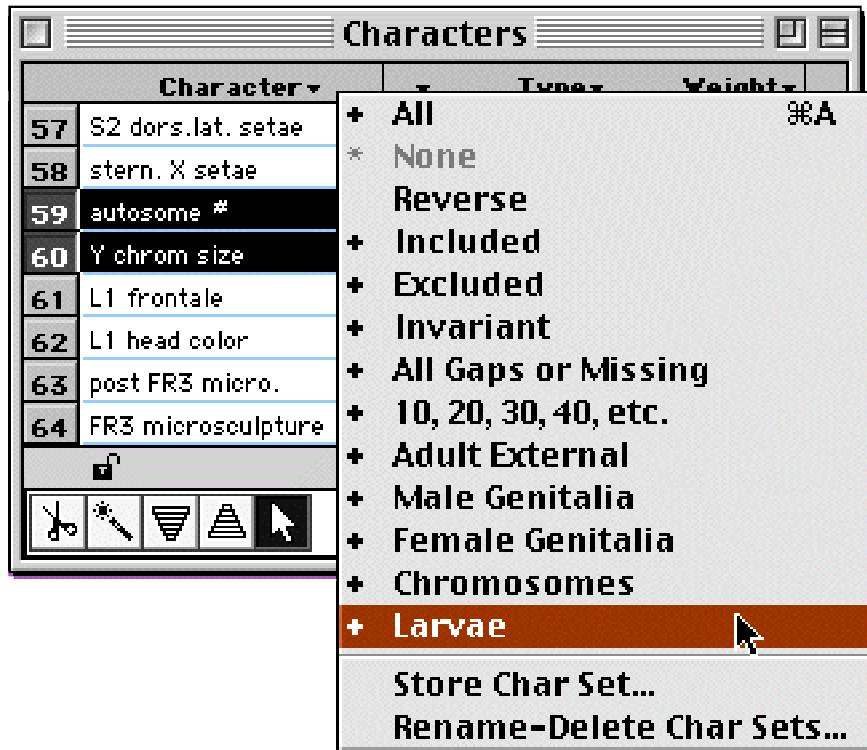
Selecting the union of two sets of objects

If you hold down the Shift or Command (⌘) key when you choose elements from the **Select** submenu, then the previously selected objects will stay selected, and the newly chosen object set will also be selected. This change in the behavior of the **Select** submenu items is indicated by the presence of a + in front of them (indicating "union").

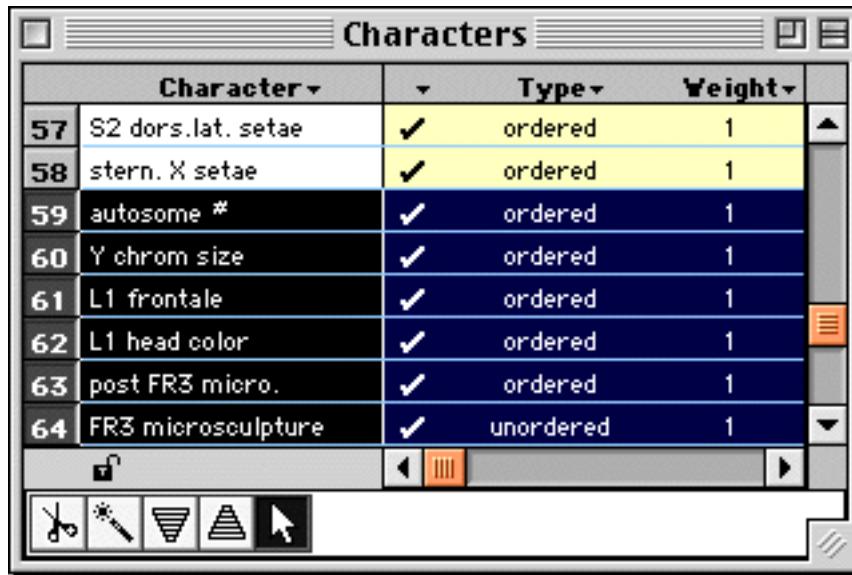
For example, imagine you wish to select all of the chromosomal and larval characters in the character list for the data matrix "Bembidion". The "Bembidion" example file on disk has several user-defined character sets, including "Chromosomes" and "Larvae". Choosing "Chromosomes" from the **Select** submenu will cause the characters in the chromosome set to be selected, and all other characters de-selected:



If you hold down the Shift key when you choose the Select submenu, +'s will appear beside the menu items, indicating that any chosen character set will be selected in addition to the already selected characters:



Choosing "Larvae" will then select the larval characters, leaving the chromosomal characters also selected:



Selecting the intersection of two sets of objects

If you hold down the Option key when you choose items from the Select submenu, then the objects selected will be those that were previously selected AND present in the chosen object set. This change in behavior of the Select menu items is indicated by a * in front of the menu items (indicating "intersection").

Selecting similar objects

MacClade allows you to select objects in list windows that have similar properties, using the Selection

Wand (). If you use the wand on any of the columns in a list window to the right of the name column, it will select all objects with the same value in that column. For example, if you touched on a "2" in the States column in the character list window:

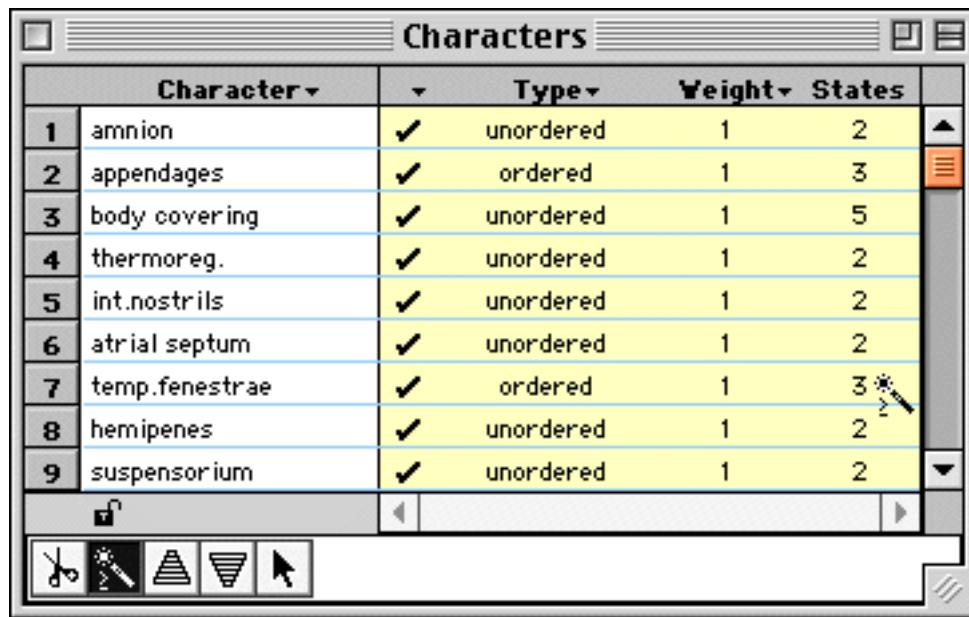
| | Character | Type | Weight | States | |
|---|----------------|-------------|--------|--------|---|
| 1 | amnion | ✓ unordered | 1 | 2 |  |
| 2 | appendages | ✓ ordered | 1 | 3 |  |
| 3 | body covering | ✓ unordered | 1 | 5 | |
| 4 | thermoreg. | ✓ unordered | 1 | 2 | |
| 5 | int.nostrils | ✓ unordered | 1 | 2 | |
| 6 | atrial septum | ✓ unordered | 1 | 2* | |
| 7 | temp.fenestrae | ✓ ordered | 1 | 3 | |
| 8 | hemipenes | ✓ unordered | 1 | 2 | |
| 9 | suspensorium | ✓ unordered | 1 | 2 |  |

then MacClade would select all objects that have two states:

| | Character | Type | Weight | States | |
|---|----------------|-------------|--------|--------|---|
| 1 | amnion | ✓ unordered | 1 | 2 |  |
| 2 | appendages | ✓ ordered | 1 | 3 |  |
| 3 | body covering | ✓ unordered | 1 | 5 | |
| 4 | thermoreg. | ✓ unordered | 1 | 2 | |
| 5 | int.nostrils | ✓ unordered | 1 | 2 | |
| 6 | atrial septum | ✓ unordered | 1 | 2 | |
| 7 | temp.fenestrae | ✓ ordered | 1 | 3 | |
| 8 | hemipenes | ✓ unordered | 1 | 2 | |
| 9 | suspensorium | ✓ unordered | 1 | 2 |  |

With this tool you can select all taxa without missing data, or all trees with polytomies, or all characters of a particular weight, and so on. Any previously selected objects will be de-selected.

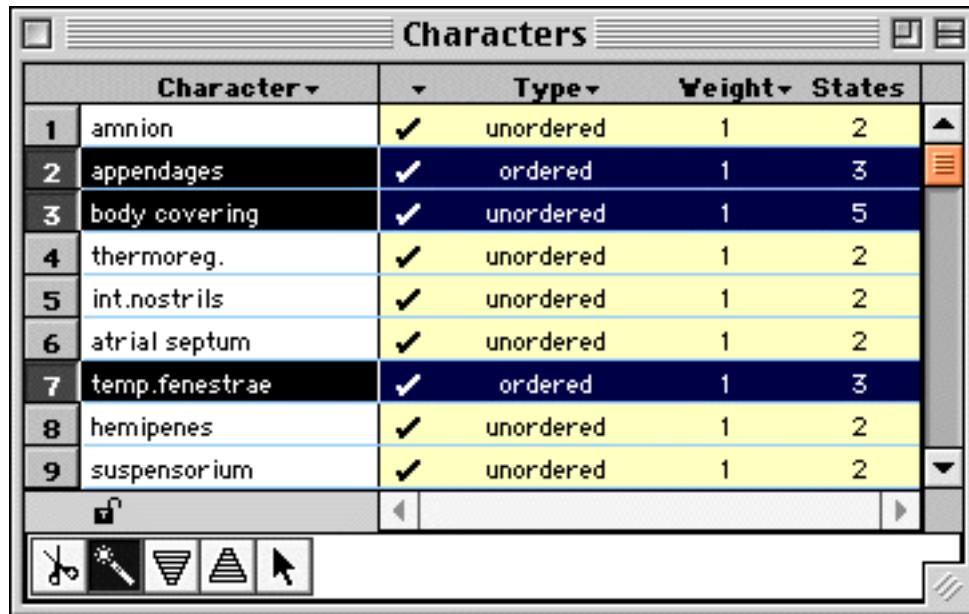
If you hold down the Option key, the wand will select all items with *greater than or equal to* that touched. For example, holding down the Option key and touching on a "3" in the States column of the character list window:



The screenshot shows the MacClade Characters window with a list of nine characters. The columns are labeled Character, Type, Weight, and States. The States column contains values 2, 3, 5, 2, 2, 2, 3, 2, and 2 respectively. A selection tool icon (a hand cursor) is positioned over the value '3' in row 7. All rows where the States value is 3 or greater are highlighted with a light blue background.

| Character | Type | Weight | States |
|------------------|-----------|--------|--------|
| 1 amnion | unordered | 1 | 2 |
| 2 appendages | ordered | 1 | 3 |
| 3 body covering | unordered | 1 | 5 |
| 4 thermoreg. | unordered | 1 | 2 |
| 5 int.nostrils | unordered | 1 | 2 |
| 6 atrial septum | unordered | 1 | 2 |
| 7 temp.fenestrae | ordered | 1 | 3 |
| 8 hemipenes | unordered | 1 | 2 |
| 9 suspensorium | unordered | 1 | 2 |

will cause all characters with three or more states to be selected:



This screenshot shows the same MacClade Characters window as the previous one, but with a different selection pattern. The selection tool icon is now positioned over the value '2' in row 8. In this configuration, rows 7 and 8 are selected, while the other rows remain unselected.

| Character | Type | Weight | States |
|------------------|-----------|--------|--------|
| 1 amnion | unordered | 1 | 2 |
| 2 appendages | ordered | 1 | 3 |
| 3 body covering | unordered | 1 | 5 |
| 4 thermoreg. | unordered | 1 | 2 |
| 5 int.nostrils | unordered | 1 | 2 |
| 6 atrial septum | unordered | 1 | 2 |
| 7 temp.fenestrae | ordered | 1 | 3 |
| 8 hemipenes | unordered | 1 | 2 |
| 9 suspensorium | unordered | 1 | 2 |

If you hold down the Control key, the Selection Wand will select all objects with values *less than or equal to* that touched.

Selecting those taxa or characters selected in the editor

If entire taxa or characters are selected in data editor, then you can ask MacClade to select the same taxa or characters in their list windows by choosing **Select Same in List Window** from the **Edit** menu. Any taxa or characters that were selected in the list window will remain selected.

Selecting all taxa with a particular state in a character

If you select a cell in the data matrix, and choose "Select all taxa with same state" in the Edit menu, MacClade will present you with the taxon list window, with all taxa with that state in that character selected. For example, with state 2 selected in character 7 in the following matrix

| | | Characters | | | 6 | 7 | 8 | |
|----|-------------|------------|-------|------|------|---|---|--|
| | | Taxa | atria | temp | hemi | | | |
| 3 | turtles | | 1 | 0 | 0 | | | |
| 4 | lungfish | | 1 | 0 | 0 | | | |
| 5 | salamanders | | 1 | 0 | 0 | | | |
| 6 | crocodiles | | 1 | 2 | 0 | | | |
| 7 | lizards | | 1 | 2 | 1 | | | |
| 8 | birds | | 1 | 2 | 0 | | | |
| 9 | mammals | | 1 | 1 | 0 | | | |
| 10 | snakes | | 1 | 2 | 1 | | | |

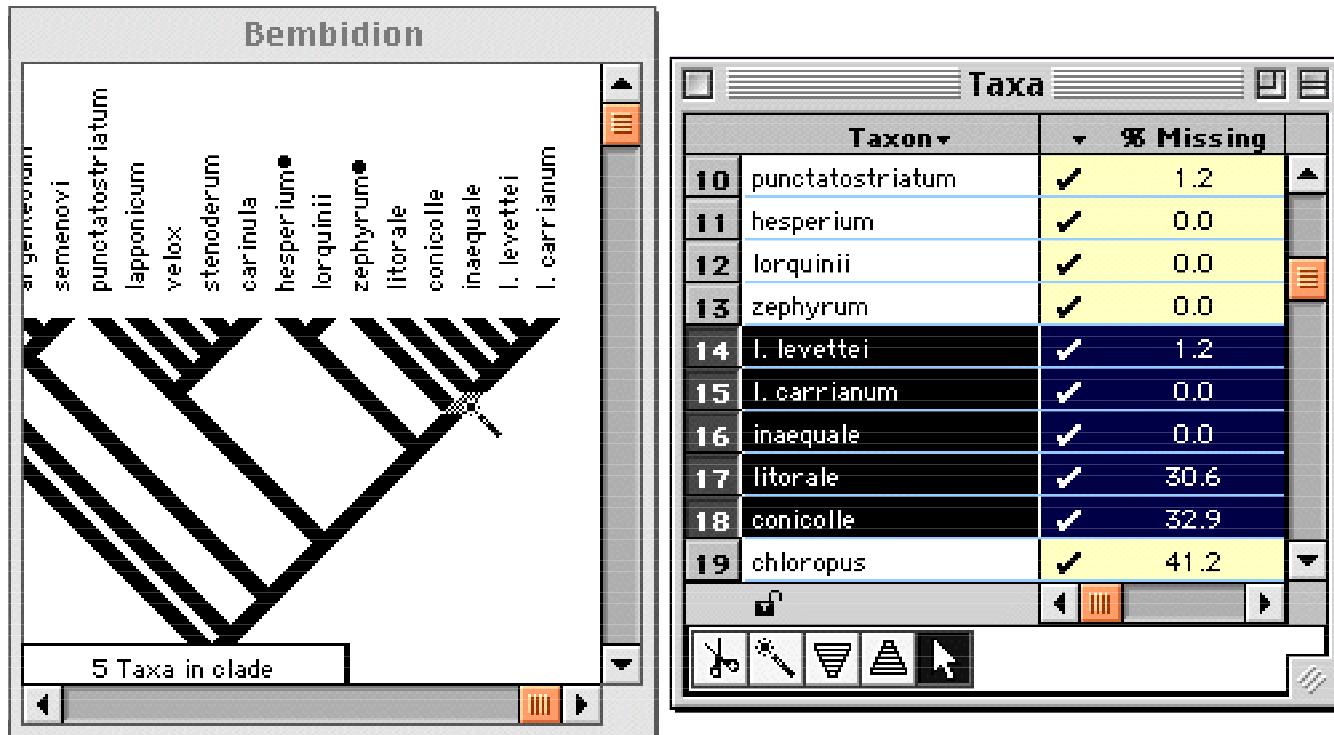
choosing "Select All Taxa with Same State" would cause crocodiles, lizards, birds, and snakes to be selected

| | | Taxon | % Missing | % Gaps | |
|----|-------------|-------|-----------|--------|--|
| 3 | turtles | | 0.0 | 0.0 | |
| 4 | lungfish | | 7.7 | 0.0 | |
| 5 | salamanders | | 0.0 | 0.0 | |
| 6 | crocodiles | | 0.0 | 0.0 | |
| 7 | lizards | | 0.0 | 0.0 | |
| 8 | birds | | 0.0 | 0.0 | |
| 9 | mammals | | 0.0 | 0.0 | |
| 10 | snakes | | 7.7 | 0.0 | |

as they all have state 2 in character 7.

Selecting all taxa in a clade

You may select all taxa in a particular clade by using the Selection Wand () from the tree window's tool palette and touching on the basal branch of the clade, as shown in the example, below:



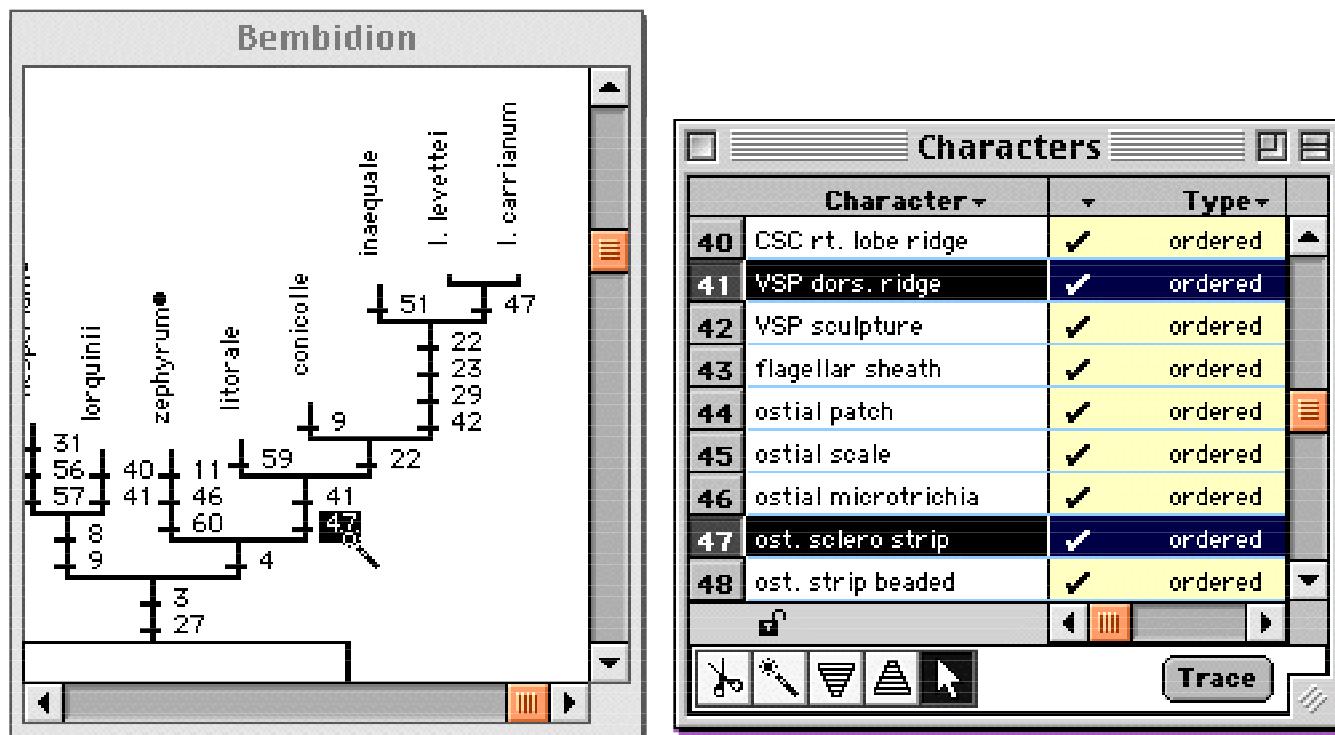
Make sure you use the selection wand chosen from the tree window's tool palette, not the Selection Wand in the list window.

If you wish to select all taxa not in a clade, you could select all taxa in the clade using the Selection Wand, and then choose **Reverse** from the taxon list window's **Select** submenu.

Selecting all characters that change along a branch

You can select all characters that are reconstructed by MacClade as changing along a branch by using the

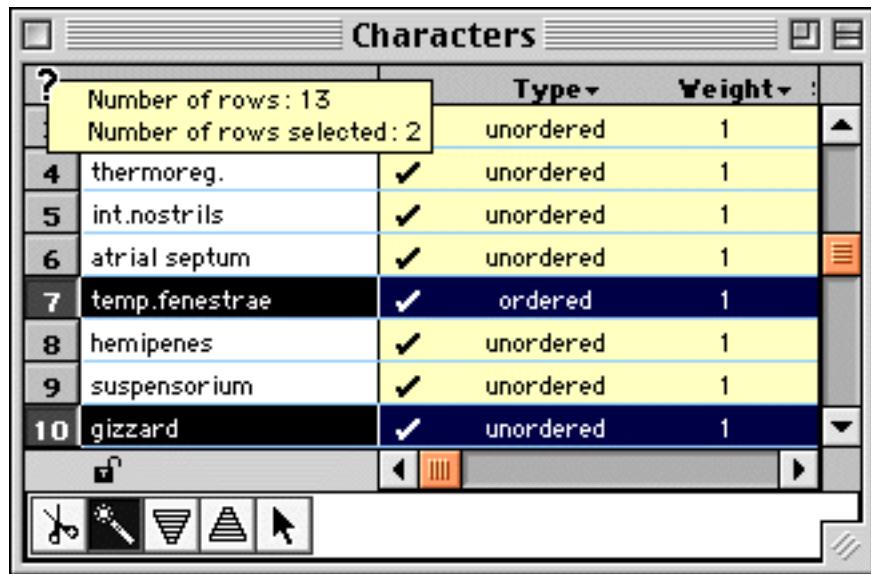
Selection Wand () from the tree window's tool palette on the branch of a tree in which Trace All Changes (see "[Summary of all changes on branches: Trace All Changes](#)" on page 358) is shown. In addition, you will need to have the changes displayed using a bar for each change (see "[Branch display](#)" on page 360). Then, if you touch on the labels for the bars, MacClade will select in the character list window all characters listed for that branch:



Make sure you use the selection wand chosen from the tree window's tool palette, not the Selection Wand in the list window.

Determining how many objects are selected

Touching on the far upper-left corner of a list window will reveal how many objects are in the list window, and how many of them are selected:

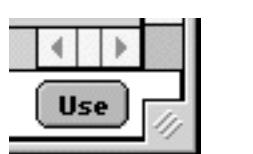


Setting values for selected characters and taxa

Once the row for a character or taxon is selected, any values for it can be set. In the character list window, you may set the character's type and weight, and whether or not it is included in analyses ([Chapter 15](#)). In the taxon list window, you can set whether or not a taxon is included in the displayed tree ("[Including and excluding taxa](#)" on page 327).

Using or displaying single selected objects

Most of the list windows have in their bottom right a button labeled "Use", "Display", or "Trace":



Type set list, weight set list,
inclusion set list



Tree list

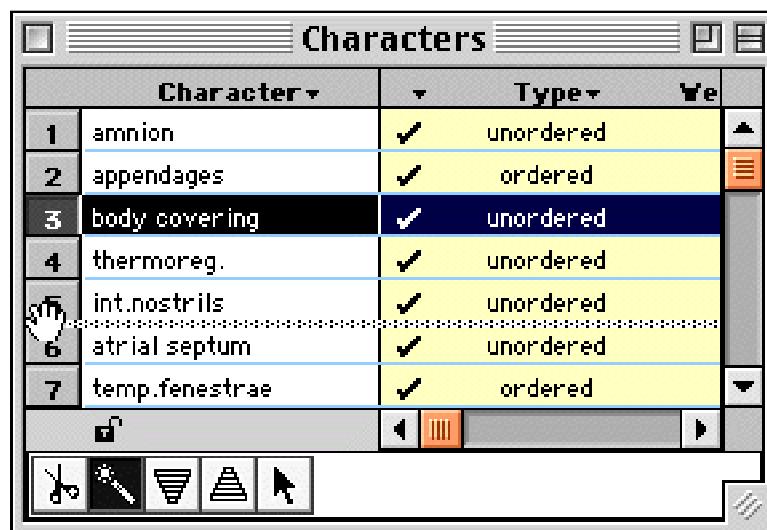


Character list

If you have a single type set, weight set, or inclusion set selected in the list, then the Use button will invoke those assumptions ([Chapter 15](#)). If you have a single tree selected in the tree list, then the Display button will display that tree in MacClade's tree window. If you have a single character selected in the character list, then the Trace button will trace that character on the tree if the tree window is open. Double-clicking on an object's row (outside of the object's name) is equivalent to selecting that row and using one of these buttons.

Rearranging the order of objects

To rearrange the order of objects in a list window, use the arrow tool, and move it over the object's number. The tool should change into a hand, and you can then drag the row to a new location. For example, grabbing the number at the left hand side of character 3 ("body covering"), and dragging it to between characters 5 and 6



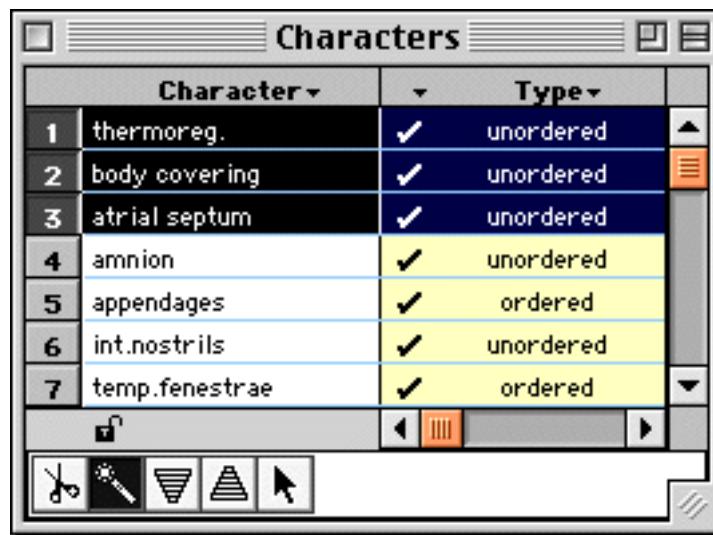
will cause the third character to be moved between characters 5 and 6:

| | Character | Type | ▼ |
|---|----------------|-------------|---|
| 1 | amnion | ✓ unordered | ▲ |
| 2 | appendages | ✓ ordered | ▼ |
| 3 | thermoreg. | ✓ unordered | ▲ |
| 4 | int.nostriis | ✓ unordered | ▼ |
| 5 | body covering | ✓ unordered | ▲ |
| 6 | atrial septum | ✓ unordered | ▼ |
| 7 | temp.fenestrae | ✓ ordered | ▲ |

You can also ask for a series of objects to be moved to the top of the list window by selecting those objects:

| | Character | Type | ▼ |
|---|----------------|-------------|---|
| 1 | amnion | ✓ unordered | ▲ |
| 2 | appendages | ✓ ordered | ▼ |
| 3 | thermoreg. | ✓ unordered | ▲ |
| 4 | int.nostriis | ✓ unordered | ▼ |
| 5 | body covering | ✓ unordered | ▲ |
| 6 | atrial septum | ✓ unordered | ▼ |
| 7 | temp.fenestrae | ✓ ordered | ▲ |

and choosing **Move Selected Rows to Top** from the **Edit** menu:



Although reordering the objects is undoable, MacClade will only undo the last action you performed. This means that if you reorder some objects, then move a branch in MacClade, or edit data, and so on, you will not be able to return the order of objects back to their original order, except by reverting to a saved version of the file. For the most part the order of objects is not critical, as it generally does not have any intrinsic meaning. However, with molecular sequence data, the order of characters is important, and for this reason MacClade will query your intentions if you try to reorder molecular sequence data.

If you wanted to store the order of a particular type of object, you could name the objects "001.", "002.", and so on, as you could then always establish the original order using the sort tools (see next section). Note that you can quickly give taxa and characters names of this sort if you use the **Fill** command in the editor (see ["Naming taxa" on page 181](#) and ["Naming characters" on page 193](#)).

There are other methods for reordering characters (see ["Reordering characters" on page 200](#)) and taxa (see ["Reordering taxa" on page 185](#)).

Sorting objects automatically

The sort tools ( and ) allow you to sort the objects in a list window in order based upon the values in a column. If you touch the sort ascending tool () on a column, MacClade will sort the objects such that the object with the lowest value is at the top, and the highest value is at the bottom. The sort descending tool () sorts in the opposite order.

For example, if you use the sort descending tool on the Steps column in the character list window

| | Character | Type | Weight | States | Steps |
|---|----------------|-------------|--------|--------|-------|
| 1 | amnion | ✓ unordered | 1 | 2 | 3 |
| 2 | appendages | ✓ ordered | 1 | 3 | 3 |
| 3 | body covering | ✓ unordered | 1 | 5 | 6 |
| 4 | thermoreg. | ✓ unordered | 1 | 2 | 2 |
| 5 | int.nostrils | ✓ unordered | 1 | 2 | 3 |
| 6 | atrial septum | ✓ unordered | 1 | 2 | 1 |
| 7 | temp.fenestrae | ✓ ordered | 1 | 3 | 7 |
| 8 | hemipenes | ✓ unordered | 1 | 2 | 2 |

the characters will be reordered such that the characters with the most steps are at the top of the list, and the characters with the fewest steps are at the bottom:

| | Character | Type | Weight | States | Steps |
|---|----------------|-------------|--------|--------|-------|
| 1 | temp.fenestrae | ✓ ordered | 1 | 3 | 7 |
| 2 | body covering | ✓ unordered | 1 | 5 | 6 |
| 3 | amnion | ✓ unordered | 1 | 2 | 3 |
| 4 | appendages | ✓ ordered | 1 | 3 | 3 |
| 5 | thermoreg. | ✓ unordered | 1 | 2 | 2 |
| 6 | hemipenes | ✓ unordered | 1 | 2 | 2 |
| 7 | suspensorium | ✓ unordered | 1 | 2 | 2 |
| 8 | gizzard | ✓ unordered | 1 | 2 | 2 |

If you use a sort tool on the column containing the names of the objects, MacClade will sort the objects alphabetically.

Deleting objects

You can permanently delete objects either by selecting their rows in the list window and then hitting the

Delete key, or touching on one of them with the list window's scissors tool (). If no rows are selected,

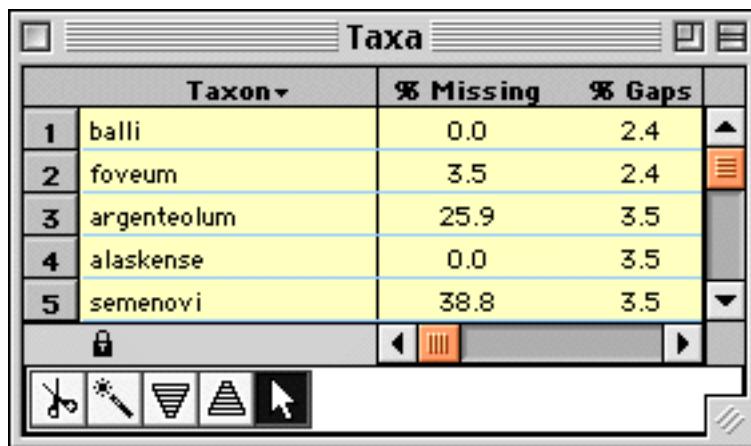
then the scissors tool will delete objects individually as you touch on them.

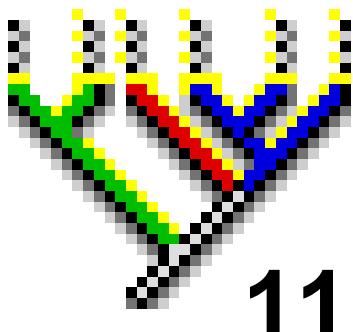
Naming objects

To rename an object in a list window, touch on the name and type in a new name:



If you wish to prevent editing of the names in a list window, touch on the lock () underneath the list of names. The lock will click shut () and you will no longer be able to edit the names until you re-open the lock by touching on it. When the names are locked, they will be yellowed to indicate you cannot edit them in the standard way:





This chapter summarizes methods for creating and manipulating taxa in MacClade.

Creating new taxa

There are three methods for adding new, empty taxa or sequences to a data matrix:

1. Using the **Add New Taxa** dialog box will add taxa to the end of the matrix.
2. Pulling the bottom edge of an untransposed matrix or right edge of a transposed matrix will add taxa to the end of the matrix.
3. Inserting rows in an untransposed matrix or columns in a transposed matrix will add taxa to the middle of a matrix.

More details, including illustrations, are found in the section ["Adding taxa or characters" on page 219](#).

Importing taxa or sequences

Although there is no mechanism in MacClade to allow easy merging of NEXUS files, or importing of additional taxa from other NEXUS files, there are features for importing molecular sequence data to an existing matrix, if those sequences are stored in the appropriate format. For details, see ["Importing sequences into an existing file" on page 147](#).

If you do wish to merge two NEXUS files of molecular sequences, the simplest method is to export one file to NBRF ([page 144](#)), and then use the importing facility to move it into the other file ([page 147](#)).

Duplicating existing taxa

To duplicate existing taxa, select the taxa in the data editor and choose **Duplicate** or **Replicate** from the **Edit** menu; see ["Making duplicates of taxa or characters" on page 220](#) for details.

Creating a consensus sequence taxon

If you have consensus sequences displayed in the editor (see ["Consensus sequences" on page 295](#)), you can ask to have a consensus sequence converted into a taxon that is stored in the data matrix itself. To do this, select the consensus sequence:

| Taxa | | Characters | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|------|-------------|------------|---|---|---|---|---|---|---|---|---|----|----|----|----|----|----|
| | Modal | | G | G | A | T | C | C | C | T | A | C | T | A | G | G | A |
| 1 | Thylacinus | | G | G | A | T | C | C | T | T | A | C | T | A | G | G | A |
| 2 | Trichosurus | | G | G | A | T | C | A | C | T | A | C | T | A | G | G | C |
| 3 | Dasyurus | | G | G | A | T | C | Y | C | T | A | T | T | A | G | G | A |

and choose **Transfer Consensus to Taxon** from the **Taxa** menu. This will move the consensus sequence to the matrix itself:

| Taxa | | Characters | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|------|-------------|------------|---|---|---|---|---|---|---|---|---|----|----|----|----|----|----|
| 1 | Modal | | G | G | A | T | C | C | C | T | A | C | T | A | G | G | A |
| 2 | Thylacinus | | G | G | A | T | C | C | T | T | A | C | T | A | G | G | A |
| 3 | Trichosurus | | G | G | A | T | C | A | C | T | A | C | T | A | G | G | C |
| 4 | Dasyurus | | G | G | A | T | C | Y | C | T | A | T | T | A | G | G | A |

Naming taxa

There are two windows in MacClade in which you can enter taxon names: the data editor, and the taxon list window.

To enter taxon names in the data editor, click on the cell for the name of the taxon in the data editor (on the left, beside the taxon numbers, for a normal matrix), and type in the name:

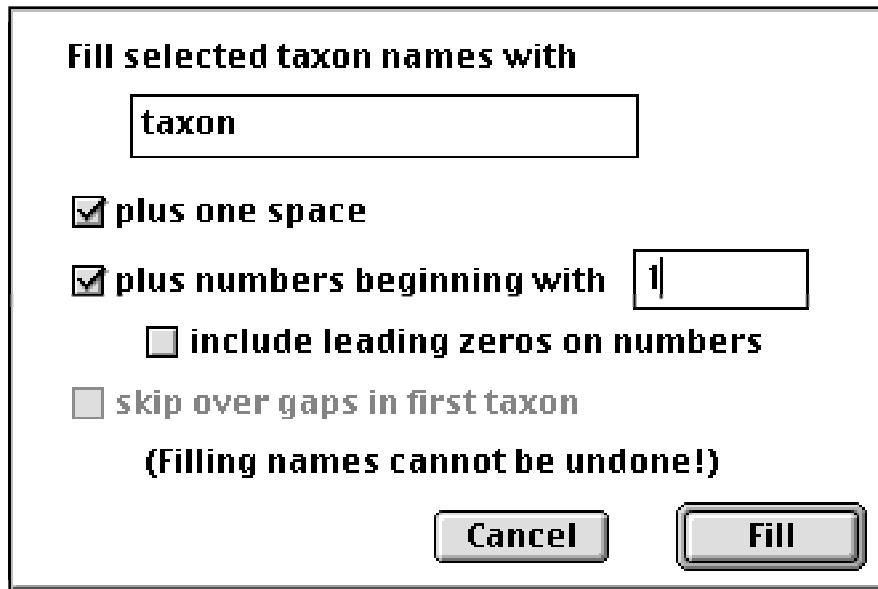
| Taxa | | Characters | 1 |
|------|----------------|------------|---|
| 1 | balli | | 1 |
| 2 | foveum* | | 1 |
| 3 | argenteolum | | 1 |
| 4 | alaskense | | 1 |
| 5 | semenovi | | 1 |
| 6 | stenoderum | | 1 |
| 7 | carinula | | 1 |

Taxon names can be up to 127 characters long; MacClade will truncate if you type in more. Any standard character is allowed in taxon names; thus valid taxon names are "Swainson's Thrush" and "Bembidion sp."

2 (western)". Taxon names cannot consist entirely of numerical digits (e.g., "23" is not allowed, but "23." is).

If you want to increase the width of the column in the data editor containing the taxon names, grab the right edge of the column and move it.

You can rapidly enter similar taxon names by selecting the block of taxon names and using the **Fill** command from the **Utilities** menu. This action is useful only if the desired taxon names form a series such as Coleoptera # 1, Coleoptera # 2, and so forth (The **Fill** command can also be used to fill character names as described in ["Naming characters" on page 193](#).) The Fill dialog box you will be presented with after you select a block of taxon names is shown below.



The Pop-up data entry tool () in the data editor provides a quick way to name taxa in a selected block of taxon names. If you touch the tool on selected taxon names in the data editor, a menu will pop up, allowing you to choose as names an alphabetic series, or the series of names "taxon 1", "taxon 2", and so on:

| Taxa | Characters | 1 | 2 | 3 | 4 | 5 |
|------|------------|-----|-------|------|---|---|
| | | eye | chead | numb | | |
| 1 | | ? | ? | ? | | |
| 2 | | ? | ? | ? | | |
| 3 | | | | | | |
| 4 | | | | | | |
| 5 | | | | | | |
| 6 | | | | | | |
| 7 | | | | | | |

A pop-up menu is displayed, showing options: "A,B,C,...", "taxon 1, taxon 2,...", and a small icon.

This might be useful for quickly naming taxa in example data files.

You can also name taxa using the taxon list window; see ["Naming objects" on page 179](#).

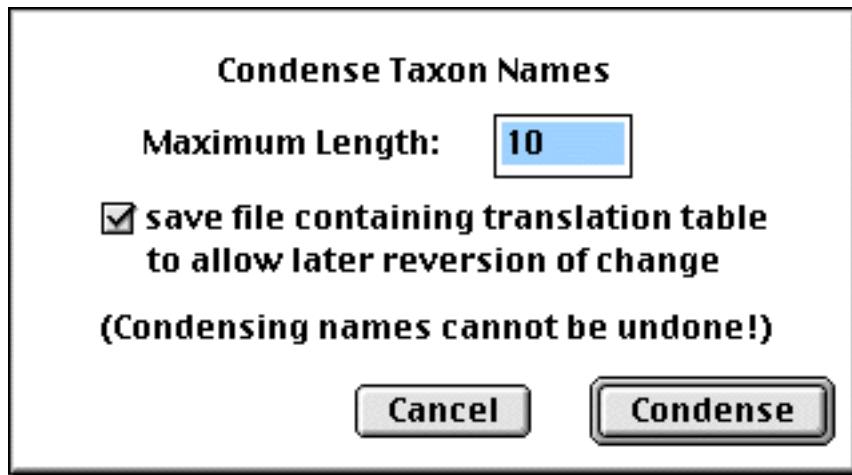
Italicizing taxon names

Taxon names can be italicized by selecting one or more taxon names in the data editor and choosing **Italic** from the **Display** menu; if the names are already italic, they will revert to plain type. Note that the whole name will become italic, not just a part of it.

Condensing and resurrecting taxon names

The **Condense Taxon Names** command in **Utilities** menu will condense the taxon names to a specified length such that each name is unique. This is useful if the file is to be exported to a program that requires unique taxon names and can only process names of a specific length.

To use this command, select the taxon names to be condensed in the data editor (see "[Selecting elements in the editor](#)" on page 210) and choose **Condense Taxon Names** in the **Utilities** menu. You will be presented with a dialog box:



By default, the names will be truncated and made unique such that all taxon names are 10 or fewer characters; you can change that number in the Condense Taxon Names dialog box. If you press the Condense button, the names will be condensed, as shown in the example, below.

| Taxa | Characters |
|------|------------------------|
| 43 | Gehrингia |
| 44 | Cymbionotum semelederi |
| 45 | Cymbionotum pictulum |
| 46 | Omophron |
| 47 | Apotomus |
| 48 | Psydrus |
| 49 | Diploous |
| 50 | Patrobus |
| 51 | Trechus |
| 52 | Diplochaetus |
| 53 | Bembidion levettei |
| 54 | Bembidion mexicanum |
| 55 | Asaphidion |

Taxon names before condensing

| Taxa | Characters |
|------|-------------|
| 43 | Gehrингia43 |
| 44 | Cymbiono44 |
| 45 | Cymbiono45 |
| 46 | Omophron46 |
| 47 | Apotomus47 |
| 48 | Psydrus48 |
| 49 | Diploous49 |
| 50 | Patrobus50 |
| 51 | Trechus51 |
| 52 | Diplocha52 |
| 53 | Bembidio53 |
| 54 | Bembidio54 |
| 55 | Asaphidi55 |

After condensing

The matrix can then be exported to another program, and the taxon names will be suitably unique.

If you wish to later resurrect the old, longer names, you will need to make sure that "save file containing translation table to allow later reversion of change" is selected in the Condense Taxon Names dialog box. If this is selected, MacClade will save a small NEXUS file containing a table indicating the correspondence between the old names and the new, shorter names. If you later have the data file open containing the condensed taxon names, and you choose **Translate Taxon Names** in the **Utilities** menu, MacClade will ask you to open up the small NEXUS file containing the translation table; it will then convert all the condensed names into their long, older counterparts.

This is valuable if the file needs to go through the "bottleneck" of another program that accepts only short taxon names. For example, imagine you wish to conduct an alignment of your data, using a program that truncates taxon names to 10 characters. The following series of steps will allow you to generate an alignment with full-length taxon names retained:

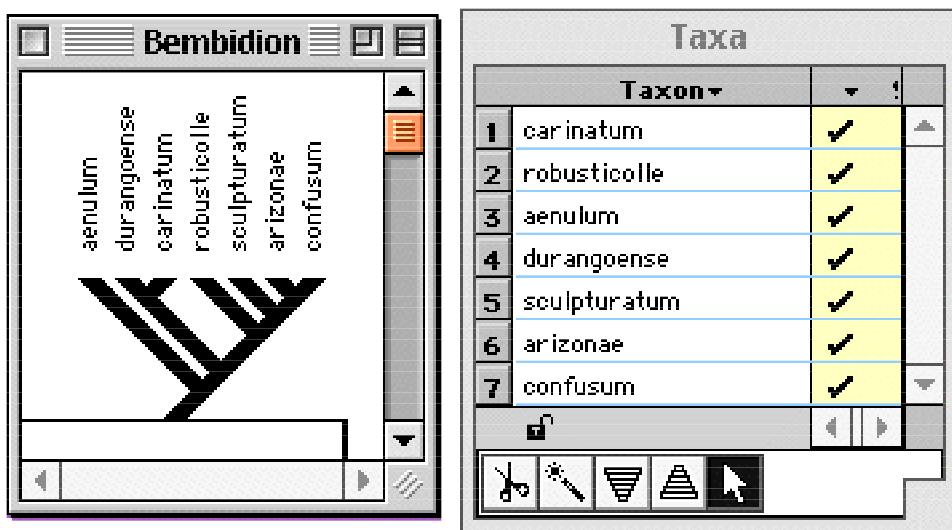
1. Accumulate sequences in a matrix, and give them full-length taxon names. Save the file.
2. When you are ready to export the sequences, first select all the taxon names by touching on one taxon name in the data editor and choosing **Select All** from the **Select** submenu of the **Edit** menu. Then choose **Condense Taxon Names** from the **Utilities** menu and press Condense. Save the translation file in some memorable place.
3. Export the file to some relevant format, for example, NBRF.
4. Read the file into the alignment program and perform the alignment, making sure it is saved in a format MacClade can read.
5. Open the resulting alignment file in MacClade.
6. Choose **Translate Taxon Names** from the **Utilities** menu and open the saved translation file.

Reordering taxa

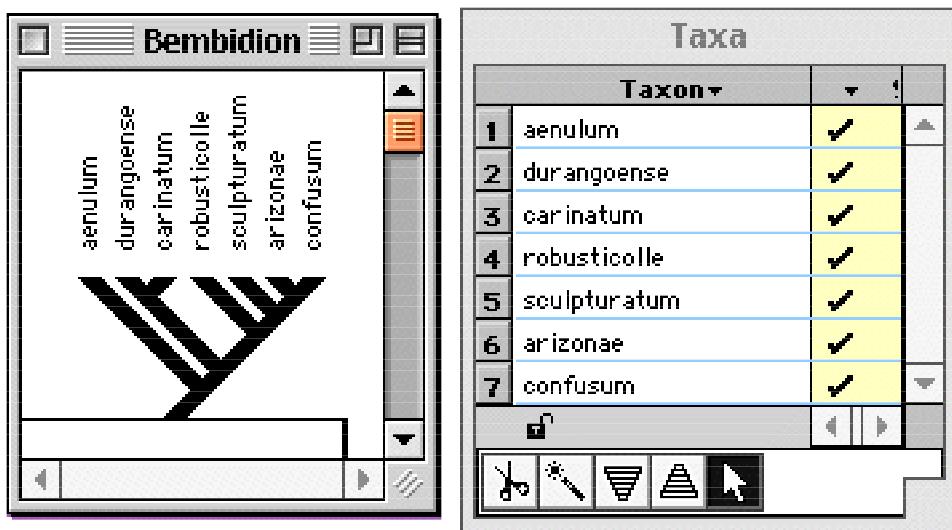
There are several methods for reordering the taxa. You can manually move the taxa to new positions in the data editor (see "[Reordering taxa or characters](#)" on page 223) or in the taxon list window ("[Rearranging the order of objects](#)" on page 175). You can also use the sorting tool in the data editor ("[Sorting tool](#)" on page 224) or in the list window ("[Rearranging the order of objects](#)" on page 175) to reorder the taxa.

In addition, you can reorder the taxa randomly using the **Reorder Randomly** command in the **Taxa** menu.

The **Reorder by Tree** command in the **Taxa** menu, available when the tree window is visible, will place the taxa in the same order that they appear (left to right) in the tree. For example, if the tree is as shown on the left and the taxa are as ordered on the right,



then choosing **Reorder by Tree** will reorder the taxa to match the order in the tree:



If some taxa are not included in the tree, they will be placed after the taxa in the current tree.

Deleting taxa from the matrix

To delete taxa completely from the data file, see the sections "[Deleting taxa, characters, and cell blocks](#)" on [page 222](#) and "[Deleting objects](#)" on [page 178](#).

Excluding and including taxa from the tree

Taxa can be presented in the data matrix but excluded from trees in the tree window. Details on excluding or including taxa in trees can be found under "[Including and excluding taxa](#)" on [page 327](#).

Merging taxa

If you select two or more taxa (the entire rows or columns must be selected), and choose **Merge Taxa** from the **Taxa** menu, MacClade will merge the taxa into one. In the process, any footnotes or pictures attached to the cells of the merged taxa will be removed. This command merges the taxon names together, and merges the character state data using the following procedures:

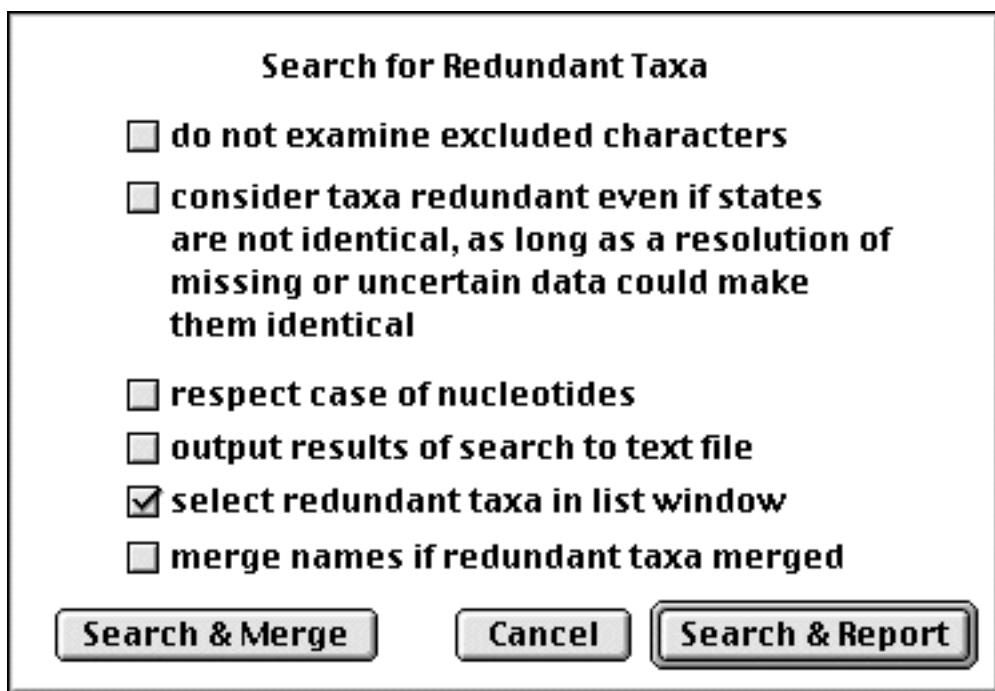
1. Merge the first two of the taxa selected, then merge this merged taxon with the next taxon selected, and so forth.
2. For each pair of taxa to be merged, MacClade does the following for each character:
 - a. If both taxa have missing data, then the merged taxon has missing data.
 - b. If one taxon has missing data, and the other has a gap, then the merged taxon has a gap.
 - c. If both taxa have gaps, then the merged taxon has a gap.
 - d. If one has missing data or a gap, but the other has neither missing data nor a gap, then the merged taxon is set to the states found in the taxon without missing data or a gap.
 - e. If both taxa are monomorphic or are polymorphic, then the merged taxon is set to the union of the states of the two taxa. If this set has more than one element, the taxon is polymorphic.
 - f. If both taxa have partial uncertainties, then the merged taxon is uncertain for the intersection of the two sets, if it exists, or for the union, if it does not (MacClade will give a warning if this happens). (Note: This procedure may yield different results depending on the order in which taxa are merged.)
 - g. If one taxa has partial uncertainty, but the other does not, then the merged taxon will take the state of the taxon with certain states. If the intersection of the two state sets does not exist, MacClade will give a warning.

These rules are exemplified by the following table:

| States of taxon 1 | States of taxon 2 | Merged states |
|----------------------|----------------------|---------------------------|
| ? | ? | ? |
| ? | - | - |
| - | - | - |
| ? | 0 | 0 |
| - | 0 | 0 |
| 0 | 0 | 0 |
| 0 | 1 | 0&1 |
| 0&1 | 0&2 | 0&1&2 |
| 0/1 | 0/1/2 | 0/1 |
| 0/1 | 2/3 | 0/1/2/3, warning given |
| 0/1 | 0 | 0 |
| 0/1 | 2 | 2, warning given |

Filtering redundant taxa

Using the **Filter Taxa** dialog box from the **Taxa** menu, you can ask MacClade to search for taxa that are redundant:



By pressing on the Search & Report button in the dialog box, MacClade will simply search for such taxa, and report on the result of the search. If you press Search & Merge, it will merge together each set of redundant taxa.

If you check the "do not examine excluded characters" box, MacClade will examine only included characters to see if the taxa are redundant; otherwise, it will examine all characters.

If you do not choose "consider taxa redundant even if states are not identical...", then MacClade will consider two taxa redundant only if their states are absolutely identical; otherwise, MacClade uses a slightly broader definition of "redundant". This broader definition is as follows. Two taxa are redundant, if, for each examined character:

1. The states of two taxa are identical, or
2. One has missing data (gaps do not count as missing data), the other does not, or
3. One and only one taxon is partially uncertain, and the states of the certain one form a subset of the uncertain one, or
4. They are both partially uncertain, and the two taxa share at least one state.

Note that if this broader definition of redundant is used, then the result of searching and merging may depend upon the order of the taxa in the matrix. Also, MacClade's Search & Report will then report not the number of redundant taxa, but the number of pairs of redundant taxa.

If you ask to filter taxa (and "consider taxa redundant even if states are not identical..." is not chosen), then MacClade will report the number of "extra" taxa. For example, if a matrix contains 4 identical taxa, MacClade will report "Number of redundant taxa: 3". If you do choose "consider taxa redundant even if states are not identical...", then MacClade will report the number of redundant pairs, in this case 6.

If "respect case of nucleotides" is checked, then MacClade will treat A as distinct from a, C from c, G from g, T from t, and U from u.

If "output results of search to text file" is checked, then MacClade will write a text file giving a detailed report of the redundant characters.

If "select redundant taxa in list window" is checked, then once the filtering is complete, MacClade will open the taxon list window, and select those taxa considered redundant with other taxa. Note that if you had asked MacClade to Search & Merge, then only one member of each redundant set of taxa would remain and be selected.

If "merge names if redundant taxa merged" is selected, then when taxa are merged, the name of the merged taxon will be a concatenation of the names of the original taxa. If this is not selected, then the name of the merged taxon will be the name of the first of the merged taxa.

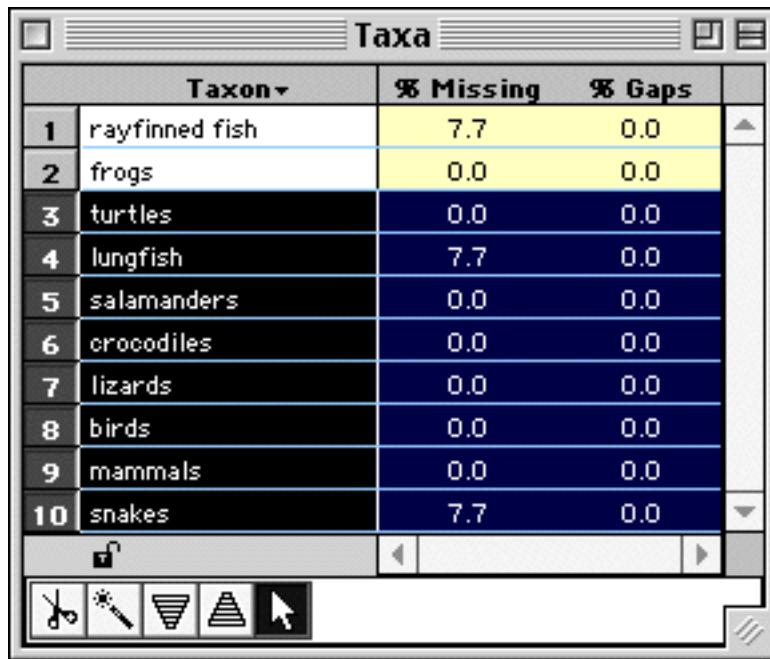
Adding notes and pictures about taxa

To add notes about individual taxa, you can either attach footnotes to their cells in the data editor (see "[Footnotes on page 443](#)") or pictures to their cells (see "[Pictures linked to taxa, character names, or data on page 444](#)"), or you can include notes about taxa in the File Notes window ([page 442](#)).

Defining your own taxon sets

Taxon sets are useful both within MacClade (as they can be used to select sets of taxa quickly in the taxon list window), and in sharing your files with other programs such as PAUP* (where taxon sets can be used to define constraint trees quickly, define outgroups, and so on).

To define a taxon set in MacClade, open the taxon list window, and select the taxa (see ["Selecting objects" on page 163](#)) you wish to include in the taxon set. In the figure below, all amniotes are selected:

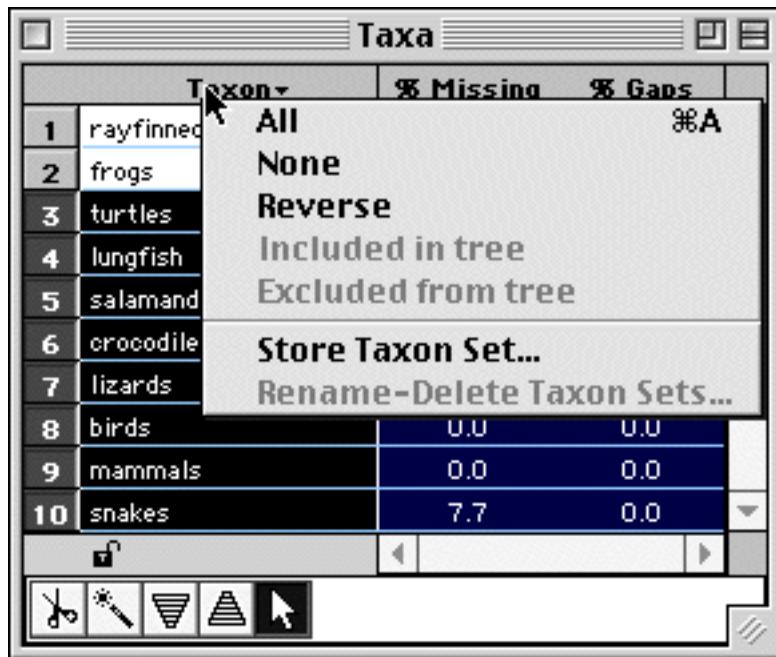


The screenshot shows the MacClade Taxa window. The title bar says "Taxa". The main area is a table with three columns: "Taxon", "% Missing", and "% Gaps". The rows are numbered 1 through 10. Rows 1, 4, and 10 have a light blue background, while the others have a dark grey background. The data is as follows:

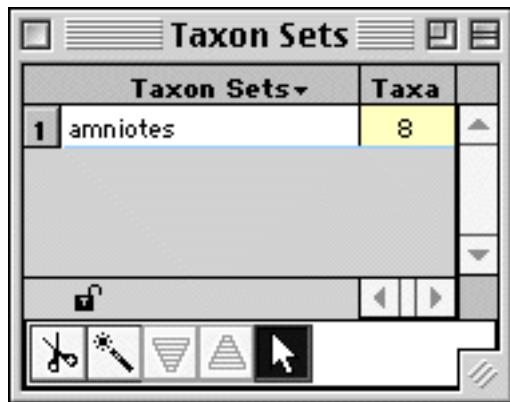
| Taxon | % Missing | % Gaps |
|------------------|-----------|--------|
| 1 rayfinned fish | 7.7 | 0.0 |
| 2 frogs | 0.0 | 0.0 |
| 3 turtles | 0.0 | 0.0 |
| 4 lungfish | 7.7 | 0.0 |
| 5 salamanders | 0.0 | 0.0 |
| 6 crocodiles | 0.0 | 0.0 |
| 7 lizards | 0.0 | 0.0 |
| 8 birds | 0.0 | 0.0 |
| 9 mammals | 0.0 | 0.0 |
| 10 snakes | 7.7 | 0.0 |

At the bottom of the window is a toolbar with icons for selection, search, filter, and other functions.

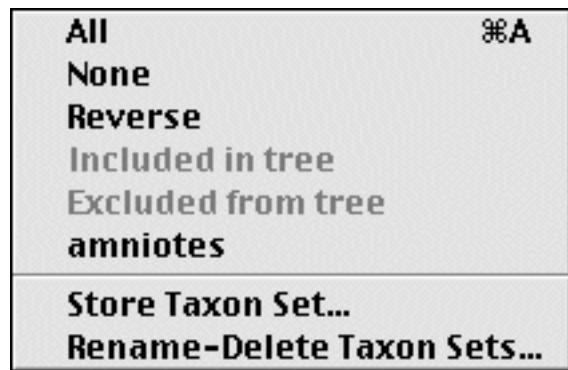
To define and store the taxon set, choose **Store Taxon Set** from either the **Select** submenu in the **Edit** menu, or from the menu that pops up when you touch on the word "Taxon" at the top of the taxon list window, as shown below:



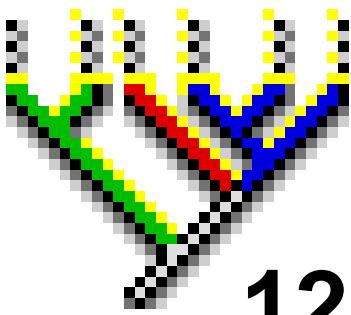
After choosing **Store Taxon Set**, the taxon sets list window will appear, and you can name the new taxon set; in the example, we might name it "amniotes":



This new taxon set is then available for use within MacClade and other programs. In MacClade, the taxon set "amniotes" will now appear within the **Select** submenu as an available taxon set to be selected:



You can rename or delete taxon sets in the taxon sets list window.



CHARACTERS AND THEIR STATES

12

This chapter summarizes the methods for creating and manipulating characters and their states within MacClade.

Data formats

The **Data Format** submenu in the **Characters** menu allows you to choose whether the data consist of standard characters, extended standard, or DNA, RNA, or protein sequences. When you first create a data file in MacClade, it is assumed to be of standard format (unless you have changed your preference for data format using the **Save Preferences** dialog box from the **File** menu), which allows 10 states per character. Extended standard format differs from standard only in allowing 26 states per character, and in so doing uses more memory. Choosing DNA or RNA format enables some features specifically designed for nucleotide sequence data, and automatically defines some symbols such as A, C, G, T, U. The protein format automatically defines as symbols the IUPAC amino acid codes. For more information about the symbols allowed by various formats, see the section ["Editing a cell" on page 224](#).

The maximum number of states allowed per character for each of the data formats is as follows:

| Format | Maximum Number of States |
|-------------------|--------------------------|
| Standard | 10 |
| Extended standard | 26 |
| DNA | 4 |
| RNA | 4 |
| Protein | 25 |

You can also include in your data file up to 10 characters with continuous values. These are edited in a separate editor, as described in [Chapter 21](#).

Changing data formats

You may find you need to change data formats after data have already been entered into the cells. For instance, you may discover you need to switch from standard to extended standard format because you find a character requiring more than 10 states. If you wish to change formats once data have already been entered into the cells, you should be aware of some limitations.

If the number of states allowed by the new format is less than that allowed by the old format, then MacClade may have to alter the data. For example, if the file was an extended standard format file, with state 15 represented for a character, and you wish to switch it to standard, then MacClade will set state 15 to missing data, as standard format can accommodate only 10 states.

If the states allowed by the new format do not include states indicated in some current user-defined trans-

formation types, then these types will be deleted. For instance, if a step matrix is defined for states 0 through 9 in a standard format data file, the type will be deleted if the data file is converted to DNA format, which allows only four states (0123 = ACGT). Any characters assigned a deleted type will be reassigned the default type.

If you wish to switch a DNA or RNA format to a protein format, then MacClade will translate the nucleotide sequence into an amino acid sequence, using the current genetic code; see ["Translating nucleotide sequences to amino acid sequences" on page 307](#) for details of this conversion.

A summary of possible conversions and their limitations follows. Note that conversion from protein to DNA or RNA data is not allowed by MacClade.

To:

| | Standard | Extended standard | DNA | RNA | Protein | |
|-------|-------------------|-------------------|-----|-----|---------|---|
| From: | Standard | ✓ | ✓ | - | - | ✓ |
| | Extended standard | - | ✓ | - | - | - |
| | DNA | ✓ | ✓ | ✓ | ✓ | * |
| | RNA | ✓ | ✓ | ✓ | ✓ | * |
| | Protein | - | - | ✗ | ✗ | ✓ |

✓: conversion acceptable

- : conversion acceptable, except some states may be removed

*: conversion will involve nucleotide to amino acid translation

✗: conversion not allowed

WARNING: If you have data stored in the Clipboard, these data are not converted to the new format until you attempt to paste them into the data editor. If you have a block of nucleotide sequence data stored in the Clipboard, these data are not translated into amino acids when you switch to protein data.

Naming characters

There are three places in MacClade in which characters can be given names: in the data editor, the State Names & Symbols window, and the character list window.

To enter the names of characters in the data editor, click on the box for the name of the character (on the top, just below the character numbers in a normal matrix; see figure, below), and type in the name. (If you need to make the columns wider to accommodate the names, you can do so using the **Column Width** submenu in the **Display** menu — see ["Changing the width of columns" on page 269](#).)

| Characters | | 1 |
|------------|---------|------|
| Taxa | | eyes |
| 1 | taxon 1 | ? |
| 2 | taxon 2 | ? |
| 3 | taxon 3 | ? |

To edit the character names in the State Names & Symbols window, click on the area to the right of the character number, and type in the name, as shown below:

| ▼ | 1. | eyes |
|---|----|------|
| 0 | 0 | |
| 1 | 1 | |
| 2 | 2 | |
| 3 | 3 | |
| 4 | 4 | |

To name characters in the character list window, use the same method as for other list windows (see "[Naming objects](#)" on page 179).

Character names can be up to 250 characters in length. Character names cannot consist entirely of numerical digits (e.g., "23" is not allowed, but "23." is).

Character names can also be entered by selecting a number of character name cells and choosing **Fill** from the **Utilities** menu (see also "[Naming sites](#)" on page 301). The **Fill** dialog box will allow you to name many characters at once. For instance, you might name 100 characters in sequence "base 200" through "base 299". If "skip over gaps in first taxon" is chosen, then MacClade will use the first taxon as a reference taxon, and not increment the numbers for those characters that have gaps in the first taxon. For example, if the first few characters of the first taxon had the following bases:

A C - - C T

and "skip over gaps in first taxon" were chosen, then MacClade could name the characters:

base 200, base 201, base 201a, base 201b, base 202, base 203.

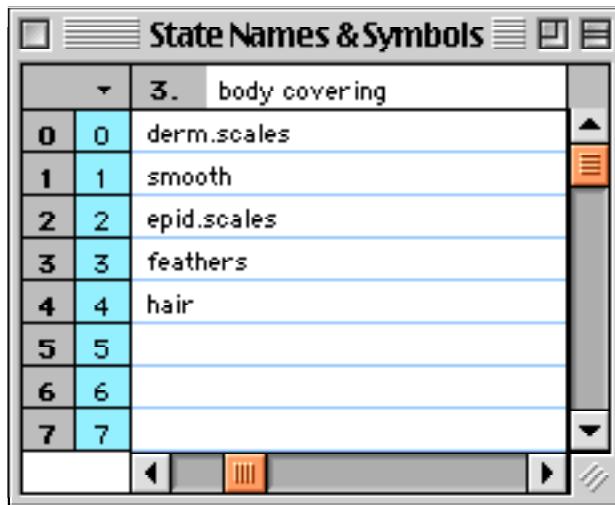
State names and symbols

State names

States are represented in MacClade by single-character symbols ("0", "1", and so on) that are used for data storage within the file; they are used throughout the program and can be displayed in the cells of the data editor. Additionally, full names (e.g., "blue", "red") can also be associated with the states of a character, and optionally can be displayed in the cells of the data editor. All characters in a matrix must share the same set of symbols; for standard and extended standard ([page 192](#)) matrices, each character can be given different state names. For more detail on the relationship between state names, symbols, and the underlying state numbers, see [page 198](#).

State names *cannot* be assigned for molecular sequence data, although three-letter amino acid names are predefined for protein matrices.

The names of character states can be added and edited only in the **State Names & Symbols** window, available in the **Characters** menu.



State names can be up to 250 characters in length. In this window, state names can be copied and pasted from one character to another by choosing **Select All** from the **Select** submenu of the **Edit** menu to select the names of all states, then copying and pasting into another character.

NOTE: *The scroll bar on the bottom of the window will take you to other characters.*

Defining state names will not necessarily cause them to be displayed in the matrix. Choose **State Names in Cells** in the **Show** submenu in the **Display** menu to ask that state names be displayed in the matrix. For protein data sets, the names of amino acids are predefined within MacClade. If you have **Three-Letter AA Names** turned on in the **Show** submenu, MacClade will display the state names in the editor.

WARNING: *Do not include within a state name the AND separator (by default "&") or the OR separator (by default "/"), as this will make it difficult for MacClade to interpret entries. If you have two states with the same name, MacClade will interpret a named entry in the matrix as referring to the smaller-valued state.*

NOTE: *If you hold down the Option key and double-click on a character name in the editor, the **State Names & Symbols** window will appear, allowing you to edit the name or state names of that character.*

If you hold down the Command key (⌘) and click on a character name in the editor, a list of the states of that character will appear with any names assigned to them.

*You can also ask MacClade to continually show you the names assigned to a selected character's states using the **Footstates** option (["Footstates" on page 272](#)).*

Symbols for states

To use symbols other than "0", "1", "2" ... for states for standard or extended standard data, choose **State Names & Symbols** from the **Characters** menu. The **State Names & Symbols** window is available only for data matrices of standard and extended standard format, because names and symbols are predefined and unchangeable for DNA, RNA, and protein formats. The cyan-colored column in the window (indicated

below with an arrow) shows the current set of symbols corresponding to each state number (the leftmost column).

| | | |
|---|---|------------------|
| | | 3. body covering |
| 0 | 0 | derm.scales |
| 1 | 1 | smooth |
| 2 | 2 | epid.scales |
| 3 | 3 | feathers |
| 4 | 4 | hair |
| 5 | 5 | |

↑
Symbols

To change the symbol used for state 1, for example, touch on the 1 in the cyan-colored box to select it:

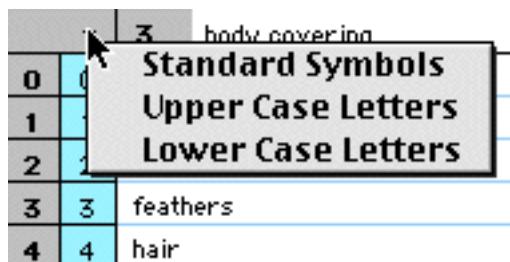
| | | |
|---|---|------------------|
| | | 3. body covering |
| 0 | 0 | derm.scales |
| 1 | 1 | smooth |
| 2 | 2 | epid.scales |
| 3 | 3 | feathers |
| 4 | 4 | hair |
| 5 | 5 | |

and then type in the new symbol for that state:

| | | |
|---|---|------------------|
| | | 3. body covering |
| 0 | 0 | derm.scales |
| 1 | a | smooth |
| 2 | 2 | epid.scales |
| 3 | 3 | feathers |
| 4 | 4 | hair |
| 5 | 5 | |

Symbols defined in this window apply to all characters.

Just above the cyan-colored column is a small triangle. A pop-up menu will appear if this is pressed, in which you can choose several other commonly used sets of symbols:



Changing symbols not only affects how the matrix is displayed, but also how the matrix is saved to the file. Changing symbols will not change the underlying data, however (as discussed in the section on state numbers, symbols, and names, below).

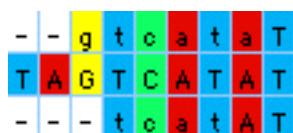
You cannot use the same symbol in more than one place (e.g., you cannot use "-" for a state symbol and the gap symbol at the same time). A blank is a valid missing data symbol. Lowercase (e.g., "a") and uppercase ("A") letter symbols are interpreted by MacClade as the same state unless you have defined the two as separate symbols in the **State Names & Symbols** window, and in some circumstances with nucleotide data (see the following section).

Symbols for character states in molecular sequence data are provided automatically when DNA, RNA, or protein formats are chosen in the **Data Format** submenu. DNA and RNA data formats come with pre-defined symbols: ACGT for DNA, ACGU for RNA. Protein data format predefines symbols to correspond to the standard IUPAC symbols for amino acids. These are:

| <u>Symbol</u> | <u>Amino Acid</u> | <u>Symbol</u> | <u>Amino Acid</u> |
|---------------|-------------------|---------------|-------------------|
| A | alanine | N | asparagine |
| C | cysteine | P | proline |
| D | aspartic acid | Q | glutamine |
| E | glutamic acid | R | arginine |
| F | phenylalanine | S | serine |
| G | glycine | T | threonine |
| H | histidine | V | valine |
| I | isoleucine | W | tryptophan |
| K | lysine | Y | tyrosine |
| L | leucine | * | chain termination |
| M | methionine | | |

Uppercase and lowercase state symbols for nucleotides

For nucleotide data, MacClade predefines the symbols ACGT or ACGU. If MacClade encounters lower-case letters (acgtu), it will treat them in all calculations exactly like their uppercase counterparts. However, it will preserve their case for display in the data editor:



and for subsequent saving to the data file.

State numbers vs. symbols vs. names

MacClade refers to states either by their *symbols* or by their *names*. Actually, these are the two modes by which MacClade refers to them in interacting with you. Internally in its own memory, MacClade refers to states simply by *numbers* (state 0, state 1, state 2, and so on) in all of its calculations, and converts state numbers to symbols or names only when it has to communicate with you. The table on [page 226](#) indicates in one example the correspondence between state numbers, symbols, and names.

Remember that the same state numbers and symbols are used for all characters in the data matrix, but state names are defined separately character by character.

For unordered characters, MacClade's internal numbering is unimportant, but for ordered characters, irreversible characters, Dollo characters, and characters of user-defined types, the internal numbering of states can be important. For instance, this internal numbering of states determines the ordering of ordered characters, and what is a gain versus loss for purposes of irreversible and Dollo characters, and so on. You can discover MacClade's internal ordering of states simply by looking at the **State Names & Symbols** window, or the **Recode** dialog box. These dialog boxes list states in the same order as is used internally by MacClade.

Technical details about numbers, symbols, and names

For the most part, you can think of MacClade as using symbols, but layering over top of the symbols, at your request, your own state names like "red" and "blue". MacClade's internal state numbering system is hidden from the user and usually need not concern you. However, it may be useful to know some basics about the relationship between numbers, symbols, and names so that you can better understand ordered characters and the effects of recoding and other actions that may affect either the data, its presentation to you, or both. This section is provided for those who want such detailed information.

Suppose you started with the following matrix of three taxa and one character, with state names "red" and "blue". Underlying these state names are (suppose) the symbols "a" and "b" which would appear if **State Names In Cells** were turned off. Underlying everything are the state numbers, which the user would normally not see except if the symbols used matched the state numbers ("0", "1", "2", ...). These state numbers determine the ordering of character states used in ordered characters, irreversible characters, and other circumstances (see ["The ordering of character states" on page 284](#)).

| As you see it | Underlying state symbols | Underlying state numbers |
|---------------|--------------------------|--------------------------|
| taxon 1..... | red | 0 |
| taxon 2..... | blue | 1 |
| taxon 3..... | red/blue | 0/1 |

Then, various actions would alter the matrix as follows:

1. Entering data, or using the **Search & Replace** dialog box ([page 239](#)), changes the state numbers MacClade stores for the cell, and therefore the symbols or names MacClade displays to you. If the original matrix above were changed by searching and replacing state "a" by a third state, "c" (whose state name happens to be "green"), then the result would be:

| As you see it | Underlying state symbols | Underlying state numbers |
|---------------|--------------------------|--------------------------|
| taxon 1 | green | c |
| taxon 2 | blue | b |
| taxon 3 | blue/gree | b/c |

2. The **Recode** dialog box ([page 238](#)) changes the state numbers, and therefore the symbols MacClade displays to you in the cell. However, because the correspondence between state numbers and state names is adjusted during recoding, it does not change the state name displayed in the cell, unless states are fused by recoding. If the original matrix above were recoded so as to change red (0=a) to 1 and blue (1=b) to 0 using the **Recode** dialog box, then the matrix would superficially appear unchanged, but the underlying data would be altered:

| As you see it | Underlying state symbols | Underlying state numbers |
|---------------|--------------------------|--------------------------|
| taxon 1 | red | b |
| taxon 2 | blue | a |
| taxon 3 | red/blue | a/b |

3. Changing the symbols list ([page 195](#)) does not change the state numbers MacClade stores for a cell, nor does it change the state names connected to those state numbers, but it does change the symbols that may appear in the cell. If the original matrix above had its symbols changed so as to invert them, making "b" the symbol for state 0 and "a" the symbol for state 1, then we would have:

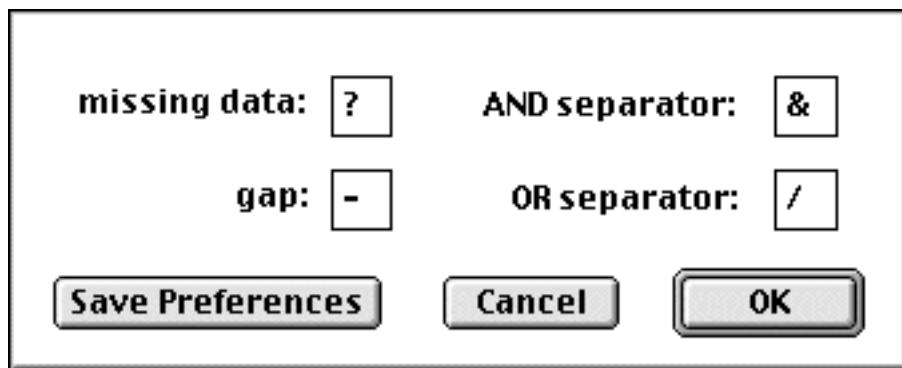
| As you see it | Underlying state symbols | Underlying state numbers |
|---------------|--------------------------|--------------------------|
| taxon 1 | red | b |
| taxon 2 | blue | a |
| taxon 3 | red/blue | a/b |

4. Changing state names ([page 194](#)) does not change the state numbers MacClade stores for a cell, nor does it change the state symbols connected to those state numbers, but it does change the state names that may appear in the cell. If the original character above had its state names changed from English to Spanish, only the appearance, not the underlying data, would change:

| As you see it | Underlying state symbols | Underlying state numbers |
|---------------|--------------------------|--------------------------|
| taxon 1 | rojo | a |
| taxon 2 | azul | b |
| taxon 3 | rojo/azul | a/b |

Symbols for missing data, gaps, AND and OR

To use symbols other than "?" for missing data, "-" for gap, "&" for the AND separator, "/" for the OR separator, choose **Missing**, **Gap**, **Multistate** from the **Display** menu when you are in the data editor, and indicate the symbol to be used for each of these.



Symbols defined in this box apply to all characters.

If you press the Save Preferences button, these symbols will be applied to future new MacClade files you create.

Creating new characters

There are three methods for adding new, empty characters to a data matrix:

1. Using the **Add New Characters** dialog box will add characters to the end of the matrix.
2. Pulling the right edge of an untransposed matrix or bottom edge of a transposed matrix will add characters to the end of the matrix.
3. Inserting columns in an untransposed matrix or rows in a transposed matrix will add characters to the middle of a matrix.

More details, including illustrations, are found in the section ["Adding taxa or characters" on page 219](#).

Duplicating existing characters

To duplicate existing characters, select the characters in the data editor and choose **Duplicate** or **Replicate** from the **Edit** menu; see ["Making duplicates of taxa or characters" on page 220](#) for details.

Reordering characters

There are several methods for reordering characters. You can manually move the characters to new positions in the data editor (see ["Reordering taxa or characters" on page 223](#)) or in the character list window (["Rearranging the order of objects" on page 175](#)). You can also use the sorting tool in the data editor (["Sorting tool" on page 224](#)) or in the list window (["Rearranging the order of objects" on page 175](#)) to reorder the characters.

Compressing characters

Choosing **Compress Characters** from the **Characters** menu will cause MacClade to search for characters with identical character states and of identical type; MacClade will then compress these identical characters into one. You must first select the characters to be examined. Excluded characters will be re-included,

reassigned their old weights, and compressed. The weight will be the sum of the weights of the compressed characters. Thus, if there were three characters (each given weight 1), all of which had A for taxa 1 and 2 but G for the remaining taxa, then MacClade would compress these three characters into one, and give that single compressed character a weight of 3.

Deleting characters from the matrix

To manually delete characters completely from the data file, see the sections "[Deleting taxa, characters, and cell blocks](#)" on page 222 and "[Deleting objects](#)" on page 178.

Deleting invariant or excluded characters

To delete invariant or excluded characters, first select them in the character list window by choosing the predefined Invariant or Excluded character set ("[Selecting or de-selecting sets of objects](#)" on page 166), and then hitting the Delete key.

For molecular sequence data, deleting characters (such as the invariable ones) can have the undesirable effect of losing the information about the original positions of characters in the sequence. MacClade remembers whether nucleotides were originally in first, second, or third positions of codons, but it does not remember original absolute position. To retain this information, you may want to number the character names automatically before deletion by selecting the character name cells of the spreadsheet and using the **Fill** command in the **Utilities** menu (e.g., "[Naming sites](#)" on page 301).

Excluding characters from analyses

To exclude or include characters from calculations, see "[Character exclusion](#)" on page 278.

Specifying assumptions about characters

For details about specifying assumptions for characters, see [Chapter 15](#).

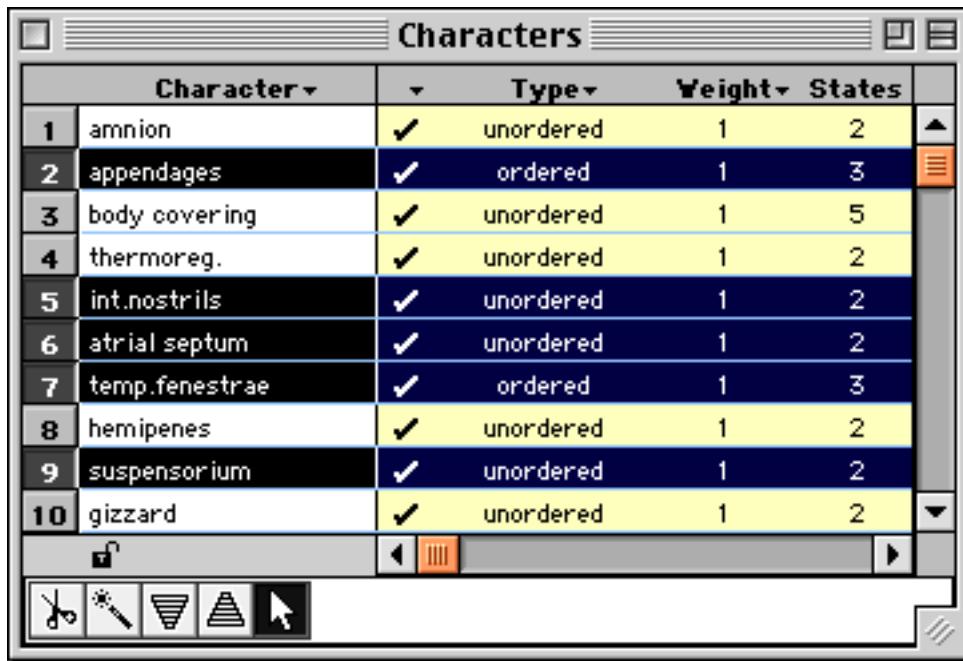
Adding notes and pictures about characters

To add notes about individual characters, you can either attach footnotes to their cells in the data editor (see "[Footnotes](#)" on page 443) or pictures to their cells (see "[Pictures linked to taxa, character names, or data](#)" on page 444), or you can include notes about characters in the File Notes window ([page 442](#)).

Defining your own character sets

Character sets are useful both within MacClade (as they can be used to select sets of characters quickly in the character list window), and in sharing your files with other programs such as PAUP* (where character sets can be used to change weights quickly, exclude characters, and so on). MacClade provides several predefined character sets, visible in the **Select** submenu of the **Edit** menu when the character list window is frontmost. You may wish to define your own character set, perhaps containing all of the characters pertaining to adults, or all to larvae, or the exons of a gene, or the stem regions in ribosomal DNA.

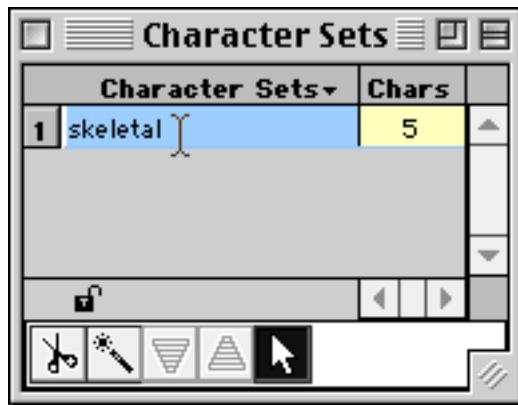
To define your own character set, select one or more characters in the character list window:



Then choose **Store Char Set** from the **Select** submenu in the **Edit** menu, or from the pop-up menu that appears when you touch on "Character" or "Site" at the top of the character list window's name column:



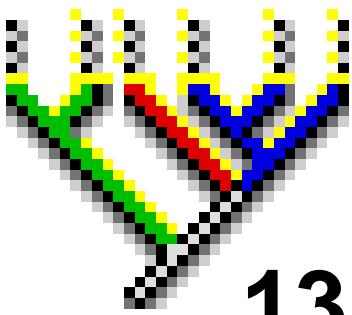
Once you choose **Store Char Set**, the character set will be stored, and the character sets list window will be presented, allowing you to name the character set:



This new character set will now appear as a new item in the **Select** submenu:



You can rename or delete character sets in the character sets list window.



ENTERING AND EDITING DATA

13

Data are entered in MacClade's data editor, a spreadsheet specialized for systematics. You can enter data by selecting a cell in the data editor and editing its contents, add, delete, or move columns and rows, have the data displayed in a number of formats, and so on. Some elements of the data editor are shown in the figure below:

The screenshot shows a MacClade data editor window titled "Vertebrates". The window has a menu bar at the top. Below the menu is a toolbar with icons for file operations. The main area is a spreadsheet grid. The columns are labeled "Taxa", "Characters", and numbered 1, 2, 3, 4. The rows are numbered 1 through 6. Column 1 contains taxon names: rayfinned fish, frogs, turtles, lungfish, salamanders, and crocodiles. Column 2 contains character names: amni, upperbody, ther. The other columns contain binary state data (0 or 1). A legend at the bottom identifies various UI elements: "taxon numbers" points to the Taxa column header; "taxon names" points to the first few entries in the Taxa column; "character numbers" points to the column headers 1, 2, 3, 4; "character names" points to the entries in the second column; "data cells" points to the individual cells in the grid; and "autotab buttons" points to the small arrows at the bottom of the grid.

| Taxa | Characters | 1 | 2 | 3 | 4 |
|------|----------------|---|---|---|---|
| 1 | rayfinned fish | 0 | 0 | 0 | 0 |
| 2 | frogs | 0 | 1 | 1 | 0 |
| 3 | turtles | 1 | 1 | 2 | 0 |
| 4 | lungfish | 0 | 0 | 0 | 0 |
| 5 | salamanders | 0 | 1 | 1 | 0 |
| 6 | crocodiles | 1 | 1 | 2 | 0 |

Before reading about the details of manipulating data in MacClade's data editor, you may wish to look at the introduction provided in the Tutorial ([page 29](#)).

Creating and editing a data matrix

To create a new data matrix you would generally follow these steps:

1. Start a new file (see [page 119](#)).
2. Choose data format (**Standard**, **Extended Standard**, **DNA**, **RNA**, or **Protein**) from the **Data Format** submenu of the **Characters** menu ([page 192](#)).
3. Expand data matrix to include the appropriate number of taxa and characters ([page 200](#)).
4. Enter taxon names ([page 181](#)).
5. Provide names for characters and states, if you wish ([page 193](#) and [page 194](#)).

6. Enter data ([page 224](#)).
7. Save data file ([page 121](#)).
8. Edit and revise data.

If you already have the data in another format, perhaps a DNA sequence alignment saved by another program, you might follow these steps:

1. Import the file into MacClade using the **Open File** menu item (see [Chapter 8](#)).
2. Import any additional single sequences you might wish to add ([page 147](#)).
3. Edit taxon names if needed ([page 181](#)).
4. Save data file ([page 121](#)).
5. Edit and revise data.

In any data file in MacClade, only two elements are necessary: names of taxa, and specification of the traits of each taxon. Optionally, you can add names of characters and character states, notes about the data, and so on.

The purpose of the data editor is to allow you to create and edit your matrix of character data. These characters are discrete-valued, that is, each character can have up to 4, 10, 25, or 26 discrete states, depending upon the data format chosen. MacClade can also store some continuous-valued characters, but these cannot be edited using the main data editor (see [Chapter 21](#) for more detail).

MacClade can display data to you in several formats. You can give names to the states of a character, and ask MacClade to display the data using those names. You can also use single-letter or number symbols. For example, if you have a DNA sequence matrix, each character has states designated by "A", "C", "G", and "T". For standard and extended standard files you can assign your own symbols. On the screen, the states can be represented therefore either by symbols (e.g., "0", "1", "2", or "A", "C", "G"), or as state names (e.g., "blue", "red").

For example, in the following matrix, the data for each of the five characters are indicated using the symbols "0", "1", and "2".

| | | Characters | | 1 | 2 | 3 | 4 | 5 |
|------|---------|------------|---|-----------|---|---|---|---|
| | | | | eye color | | | | |
| Taxa | | 1 | 2 | 3 | 4 | 5 | | |
| 1 | taxon 1 | 0 | 2 | 1 | 0 | 1 | | |
| 2 | taxon 2 | 0 | 1 | 2 | 0 | 0 | | |
| 3 | taxon 3 | 1 | 1 | 1 | 1 | 2 | | |
| 4 | taxon 4 | 0 | 1 | 2 | 2 | 1 | | |
| 5 | taxon 5 | 1 | 1 | 0 | 2 | 1 | | |

Note that the third character has been given a name.

If we then name the states of the third character (such that state 0 is "red", state 1 is "blue", and state 2 is "green"; ["State names" on page 194](#)), and ask MacClade to display the information using state names, we

see that the third character is now shown using state names:

| Taxa | | Characters | | 1 | 2 | 3 | 4 | 5 |
|------|---------|------------|--|-----------|---|-------|---|---|
| | | | | eye color | | | | |
| 1 | taxon 1 | | | 0 | 2 | blue | 0 | 1 |
| 2 | taxon 2 | | | 0 | 1 | green | 0 | 0 |
| 3 | taxon 3 | | | 1 | 1 | blue | 1 | 2 |
| 4 | taxon 4 | | | 0 | 1 | green | 2 | 1 |
| 5 | taxon 5 | | | 1 | 1 | red | 2 | 1 |

If we change the symbols used by MacClade for the states to a, b, and c (see "[Symbols for states](#)" on [page 195](#)), then all characters (except those with defined state names) are displayed using those new symbols:

| Taxa | | Characters | | 1 | 2 | 3 | 4 | 5 |
|------|---------|------------|--|-----------|---|-------|---|---|
| | | | | eye color | | | | |
| 1 | taxon 1 | | | a | c | blue | a | b |
| 2 | taxon 2 | | | a | b | green | a | a |
| 3 | taxon 3 | | | b | b | blue | b | c |
| 4 | taxon 4 | | | a | b | green | c | b |
| 5 | taxon 5 | | | b | b | red | c | b |

The underlying data, as stored within MacClade's memory, are the same for each of these three displays. For full details on the relationships between numbers, symbols, and state names, see the section "[State numbers vs. symbols vs. names](#)" on [page 198](#).

Getting to the data editor

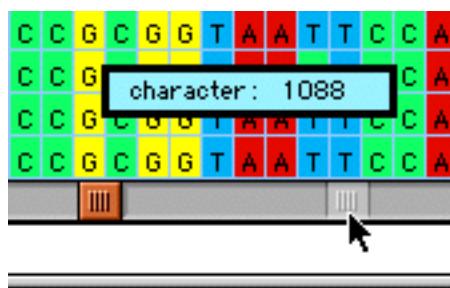
MacClade will automatically open the data editor when a new file is created; this allows you to begin entering data in the new matrix. If you import data from another format MacClade will first present you with the data in the data editor. If you are working with a previously saved file, MacClade will automatically open the data editor if the file was last saved with the editor window open, or if you hold the Option key down while the file is being read.

If you are in the tree window and wish to move to the data editor window, or the editor is hidden under another window and you wish to bring the editor to the front, choose **Data Editor** from the **Windows** menu.

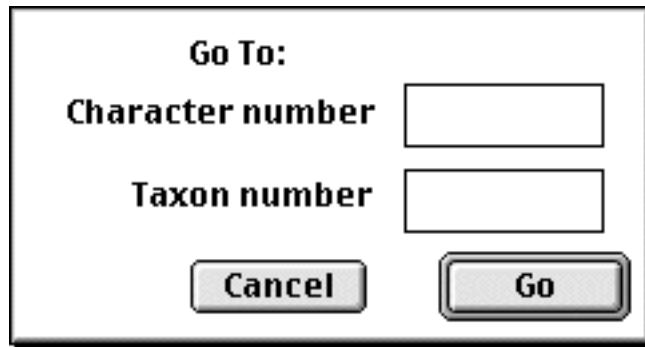
Moving around the matrix

If the matrix is too large to be completely visible in the editor, you can scroll through the matrix using the scroll bars on the right and bottom of the editor. If you grab the scroll bar's thumb to move to another spot in the matrix, MacClade will give you feedback as to the position in the editor of that point along the scroll

bar:



To move quickly to a particular spot in the matrix, you can use the **Go To** submenu of the **Edit** menu. If part of the matrix is selected, then choosing **Selection** will move the selection into view. Choosing **Home** will move the matrix so that character 1 and taxon 1 is at the upper left. Choosing **Taxon & Character** will call forth a dialog box in which you enter the number of a character or taxon:

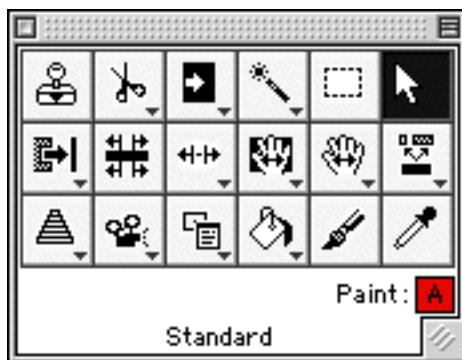


to which the editor will move.

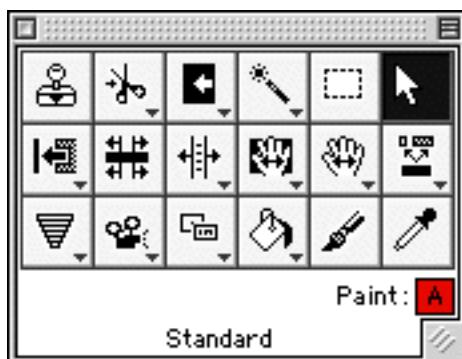
If **Track Cell** in the **Edit** menu is checked, then MacClade will keep the currently selected cell in view. If a cell is selected, but not visible, and you enter text into it, or you use the arrow keys or autotab to move to a cell that is out of view, MacClade will automatically scroll the matrix until the cell is visible. If **Track Cell** is not checked, then MacClade will not automatically scroll the matrix in this manner.

Data editor tools

The data editor has a selection of tools available in the tool palette ([page 158](#)), which can be used to select, manipulate, or display data. The standard tool palette is:



When the Option key is held down, some of the tools switch to an alternative mode:



For a general introduction to use of a tool palette, including choice of options using the tools' pop-up menus, see ["Tool palettes" on page 158](#).

The standard tools, and locations in this manual of descriptions of their actions, are as follows:

| Tool | Description |
|------|--|
| → | Standard editor tool. If another tool is currently selected, you can get back to this tool by holding down the Command (⌘) key. |
| □ | Block Selector. Dragging this tool over the matrix allows you to select a block of cells. See page 213 . |
| ★ | Selection Wand. Selects all taxon or character names, or selects blocks of cells bounded by gaps or nongaps. See page 213 . |
| ► | Select-to-end or Select-to-start tool. Selects from the cell touched to the end or start of the matrix. See page 217 |

| Tool | Description |
|---|--|
|  | Scissors. Excises the selected block of cells from the matrix. See page 222 . |
|  | Stamp Clipboard tool. Pastes a block of cells contained in the clipboard into the matrix at the position touched. See "Stamp clipboard tool" on page 237 . |
|  | Pairwise alignment tool. Conducts pairwise alignment of one sequence onto another. See "Pairwise alignment tool" on page 247 . |
|  | Block Mover. Moves a sequence or a portion of a sequence when dragged. See "Block Mover" on page 242 . |
|  | Selected Block Mover. Moves a selected block when dragged. See "Selected Block Mover" on page 243 . |
|  | Block Splitter. Splits one or more sequences, introducing gaps. See "Block Splitter" on page 244 . |
|  | Split others tool. Introduces gaps in all sequences except the one being dragged. See "Split Others tool" on page 245 . |
|  | Close gaps tool. Moves a selected block toward the start or end as far as possible through gaps; can also remove gaps in a selected block. See "Close Gaps tool" on page 246 . |
|  | Choose state tool. Chooses the state of the "paint" used by the paint states and fill states tools. See "Choose state (Eyedropper) tool" on page 231 . |
|  | Paint states tool. Sets the contents of data cells to that represented by the "paint". See "Paint states tool" on page 232 . |
|  | Fill states tool. Fills a block of cells with the states represented by the "paint" or with random data. See "Fill states tool" on page 233 . |
|  | Pop-up data entry. Fills a cell or block of cells with states chosen from a pop-up menu, or allows you to see the full contents of a cell. See "Pop-up data entry tool" on page 234 . |

| Tool | Description |
|---|--|
|  | Show picture tool. This tool displays any images attached to cells in the matrix. See " "Show Picture tool" on page 446 . |
|  | Sort tool. This tool sorts the characters or taxa. See " "Sorting tool" on page 224 . |

Treatment of missing data by the editor's data manipulation tools

Some of the tools in the data editor, especially those used for molecular sequence alignment, treat gaps very differently than data in other cells. These tools, by default, treat cells with missing data not as gaps, but as data. You can change this behavior so that missing data is treated by these tools as equivalent to gaps.

For example, the Block Mover ([page 242](#)) automatically defines as a block of sequence data a contiguous string of data (nongaps) bounded on each end by gaps (or by the end of the matrix). For example, if you applied the Block Mover to the following piece:

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| - | - | - | T | T | C | G | G | - | C | G | - | - |
| - | - | - | T |  T | G | C | ? | T | A | - | - | - |
| - | - | - | T | T | C | G | G | - | C | G | - | - |

then the block that would be moved would include TTTCGC?TA:

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| - | - | - | T | T | C | G | G | - | C | G | - | - |
| - | - | - | T |  T | G | C | ? | T | A | - | - | - |
| - | - | - | T | T | C | G | G | - | C | G | - | - |

However, if you choose **Editor Tools Treat Missing as Gap** in the **Edit** menu, then the block would not include the cell with missing data, and would thus only include TTTCGC, as the ? would be treated as equivalent to a gap:

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| - | - | - | T | T | C | G | G | - | C | G | - | - |
| - | - | - | T |  G | C | ? | T | A | - | - | - | - |
| - | - | - | T | T | C | G | G | - | C | G | - | - |

This affects all tools in the editor, which then treat gaps differently from nucleotides or amino acids.

Selecting elements in the editor

Selecting individual cells

To select an individual cell, so that you may enter data into it, just click on the cell with the arrow tool. The text in the cell will be selected.

| Characters | | 1 | 2 | 3 | 4 |
|------------|---------|-----------|---|-------|---|
| Taxa | | eye color | | | |
| 1 | taxon 1 | a | c | blue | |
| 2 | taxon 2 | a | b | green | |
| 3 | taxon 3 | b | b | blue | |
| 4 | taxon 4 | a | b | green | |
| 5 | | | | | |

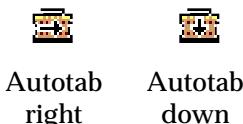
Selecting a cell

Just as with most Macintosh text editing, if you type something with the whole cell selected, the cell's contents will be replaced by what you type. If you click on the cell again, you will put the blinking vertical bar (the insertion point) into the cell where you click. The usual Macintosh rules apply to other aspects of editing within a cell, such as placing the insertion point, selecting pieces of text within the cell, and entering new text.

Moving the selection from cell to cell

You can move to other cells by clicking on them using the pointer, or to adjacent cells using the arrow keys. Alternatives to the arrow keys are the Return and Tab keys. The Return key moves to the next cell lower in the column, the Tab key moves to the next cell right in the row. Shift-Return and Shift-Tab move to the next cell above and left, respectively.

To have MacClade move automatically to the next cell after you have typed in a state, select one of the two autotab buttons at the bottom left of the window.



If autotab is enabled, then just after you type something into a cell, the next cell will be selected automatically. If you have the button with the right-pointing arrow selected, the cell to the right will be selected; if you have the other one selected, the cell below. When the end of a row or column is reached, MacClade will beep and return to the start of the next row or column. This feature facilitates rapid entry of data. However, while in this mode you can type only one symbol per cell.

Selecting rows, columns, and blocks

Whole rows and columns can be selected by clicking on the number of the row or column:

| 1 | 2 |  | 4 | 5 |
|-----------|---|---|---|---|
| eye color | | | | |
| a | c | blue | b | a |
| a | b | green | a | b |
| b | b | blue | b | a |
| a | b | green | c | b |
| a | b | blue | b | b |
| a | b | green | c | a |

Selecting a whole column

You can select several rows or columns using the standard Shift-click to extend the selection. Click on the first row or column you would like selected, release the mouse button, hold down the Shift key, and touch on the last row or column you would like selected. You *cannot* make a discontinuous selection using ⌘-click as you can, for example, in the character list window.

You can also select whole columns or rows using the Macintosh standard Shift-click technique to extend a selection: Click on the name or first cell in the row or column, hold down the Shift key, and click on the last cell in the row or column.

You can select a block of the matrix with Shift-click using the standard tool. For example, to select characters 3 through 7 of taxa 3 to 8, click once on the cell for taxon 3, character 3. Let go of the mouse button, and, holding down the Shift key, click once on the cell of taxon 8, character 7. The result should be a selected block, as shown below.

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|------|-------|------|------|------|-------|------|-----|------|
| amni | upper | body | ther | int. | natri | temp | hem | susp |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ? |
| 0 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 |
| 1 | 1 | 2 | 0 | 1 | 1 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | ? |
| 0 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 |
| 1 | 1 | 2 | 0 | 1 | 1 | 2 | 0 | 0 |
| 1 | 1 | 2 | 0 | 1 | 1 | 2 | 1 | 1 |
| 1 | 2 | 3 | 1 | 1 | 1 | 2 | 0 | 0 |
| 1 | 1/2 | 4 | 1 | 1 | 1 | 1 | 0 | 0 |
| 1 | ? | 2 | 0 | 1 | 1 | 2 | 1 | 1 |

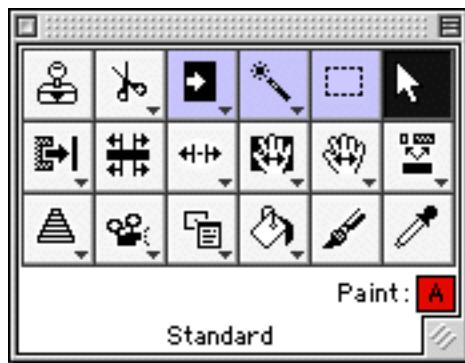
You can also select a block using the Block Selector ( , see below), by clicking and dragging to expand the selected area.

If a cell or block of cells is selected in the editor, you can also extend the selection using Shift-Arrow key (Shift →, Shift ↓, Shift ←, or Shift ↑). This will extend the selection one row or column over in the direction of the arrow. For example, typing Shift → when the block pictured above is selected will extend the selection to include taxa 3 through 8 of character 8. Shift-Option-Arrow key will extend the selection to the edge of the matrix.

You can select the entire matrix quickly by choosing **Select All** from the **Select** submenu in the **Edit** menu; you can de-select all cells by choosing **None** from the **Select** submenu in the **Edit** menu.

Editor tools for selecting data

Three tools in the editor, shaded blue in the figure below, will select blocks of cells in the editor.



Block selector

Touching this tool on the matrix and then dragging it over a block of cells will select that block. For example, if you touched on the cell of taxon 2 and character 2 (left figure, below) and then dragged the tool toward the lower right, a block of cells would be selected (right figure, below).

| | | | | | |
|---|---|---|---|---|---|
| ? | 2 | ? | ? | 0 | ? |
| 1 | 2 | 0 | 1 | 1 | 3 |
| ? | 2 | ? | ? | 0 | ? |
| 1 | 2 | 0 | 1 | 0 | 1 |
| 1 | 2 | 1 | 0 | 2 | 0 |
| 0 | 2 | 2 | 0 | 0 | 0 |
| 0 | 2 | 2 | 0 | 0 | 0 |

| | | | | | |
|---|---|---|---|---|---|
| ? | 2 | ? | ? | 0 | ? |
| 1 | 2 | 0 | 1 | 1 | 3 |
| ? | 2 | ? | ? | 0 | ? |
| 1 | 2 | 0 | 1 | 0 | 1 |
| 1 | 2 | 1 | 0 | 2 | 0 |
| 0 | 2 | 2 | 0 | 0 | 0 |
| 0 | 2 | 2 | 0 | 0 | 0 |

Using the Block Selector to select a block of cells



Selection Wand

The action of the Selection Wand varies, depending on what is touched.

If you touch on a taxon name, then all taxon names will be selected:

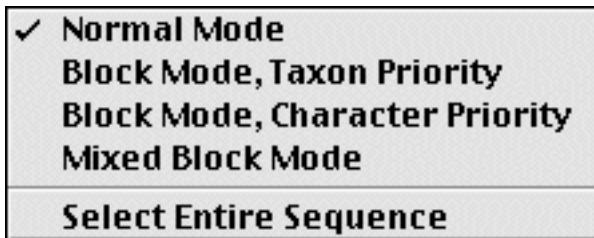
| Taxa | | Characters | | 1 | 2 | |
|-----------|----------------|------------|-----|------|-------|--|
| | | | | amni | apper | |
| 1 | rayfinned fish | 0 | 0 | | | |
| 2 | frogs | 0 | 1 | | | |
| 3 | turtles | 1 | 1 | | | |
| 4 | lungfish | 0 | 0 | | | |
| 5 | salamanders | * | 1 | | | |
| 6 | crocodiles | 1 | 1 | | | |
| 7 | lizards | 1 | 1 | | | |
| 8 | birds | 1 | 2 | | | |
| 9 | mammals | 1 | 1/2 | | | |
| 10 | snakes | 1 | ? | | | |
| 11 | | | | | | |

| Taxa | | Characters | | 1 | 2 | |
|-----------|----------------|------------|-----|------|-------|--|
| | | | | amni | apper | |
| 1 | rayfinned fish | 0 | 0 | | | |
| 2 | frogs | 0 | 1 | | | |
| 3 | turtles | 1 | 1 | | | |
| 4 | lungfish | 0 | 0 | | | |
| 5 | salamanders | * | 1 | | | |
| 6 | crocodiles | 1 | 1 | | | |
| 7 | lizards | 1 | 1 | | | |
| 8 | birds | 1 | 2 | | | |
| 9 | mammals | 1 | 1/2 | | | |
| 10 | snakes | 1 | ? | | | |
| 11 | | | | | | |

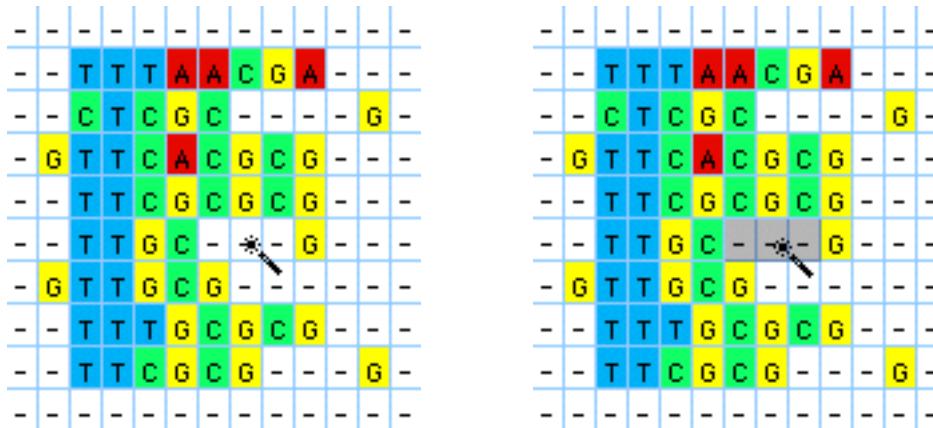
Touching the wand on a taxon name will select all taxon names

In a similar fashion, touching the wand on a character name will select all character names.

If you touch on a data cell, the tool will select a block of cells, with the boundary being determined by the distribution of gaps. (Missing data will be treated as data, not gaps, unless you have selected Editor Tools Treat Missing As Gap, [page 210](#).) This was designed for use with molecular sequence data. In general, if you touch on a cell that is a gap, then the largest contiguous piece of pure gaps containing that cell is selected; if you touch on a non-gap, then the largest contiguous piece of nongaps is selected. The exact behavior is a function of the tool's mode, as selected in the tool's pop-up menu:

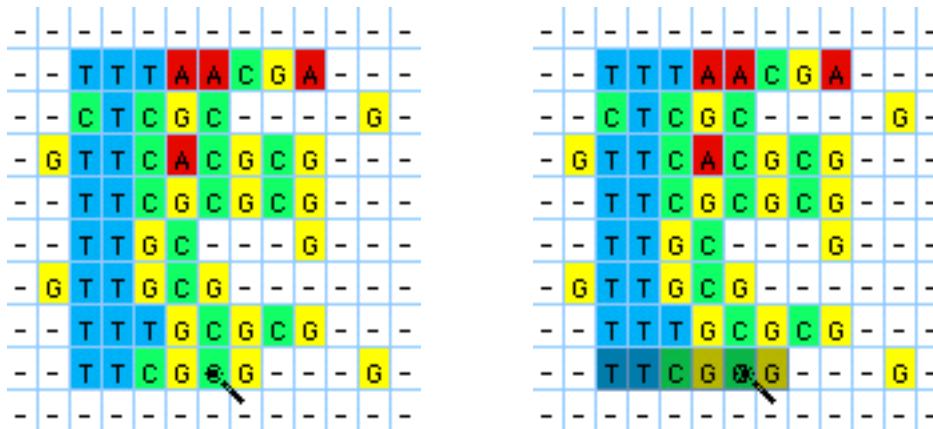


In **Normal Mode**, the tool selects a part of a single sequence. If you touch on a gap, then a series of contiguous gaps in that sequence will be selected:



Touching the wand in normal mode on a gap

Similarly, if you touch on a cell containing something other than a gap (i.e., a nucleotide), then a contiguous block of nongaps in that sequence will be selected:



Touching the wand in normal mode on a cell containing data

If **Block Mode, Taxon Priority** is chosen, MacClade attempts to select the largest block of like kind (that is, containing no gaps or containing only gaps), maximizing the number of taxa selected. If you touch on a cell containing data, then MacClade will select a block of cells by first examining cells in adjacent taxa, and looking for the contiguous block of taxa with data at that character (below, left); this determines the extent of the final selection across taxa. Then, with the taxon extent determined, MacClade scans across characters for the largest span of characters that would produce a block containing only data (below, right).



In this example, the final selection would be:

| | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| - | - | T | T | T | A | A | C | G | A | - | - | - | - | - | - | - | - | - | - | - | - | - |
| - | - | C | T | C | G | C | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| - | G | T | T | C | A | C | G | C | G | - | - | - | - | - | - | - | - | - | - | - | - | - |
| - | - | T | T | C | G | C | G | C | G | - | - | - | - | - | - | - | - | - | - | - | - | - |
| - | - | T | T | G | C | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| - | G | T | T | G | C | G | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| - | - | T | T | T | G | C | G | C | G | - | - | - | - | - | - | - | - | - | - | - | - | - |
| - | - | T | T | C | G | O | G | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |

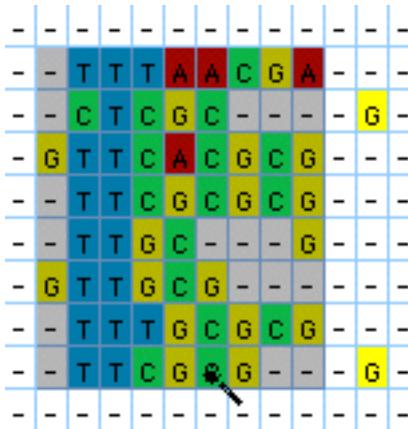
If **Block Mode, Character Priority** is chosen, MacClade attempts to select the largest block of like kind (that is, containing no gaps or containing only gaps), maximizing the number of characters selected. If you touch on a cell containing data, then MacClade will select a block of cells by first examining cells in adjacent characters, and looking for the contiguous block of characters with data at that taxon (below, left); this determines the extent of the final selection across characters. Then, with the character extent determined, MacClade scans across taxa for the largest span of taxa that would produce a block containing only data (below, right).



In this example, the final selection would be:

| | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| - | - | T | T | T | A | A | C | G | A | - | - | - | - | - | - | - | - | - | - | - | - | - |
| - | - | G | T | T | G | C | G | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| - | - | T | T | T | G | C | G | C | G | - | - | - | - | - | - | - | - | - | - | - | - | - |
| - | - | T | T | G | C | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| - | G | T | T | G | C | G | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| - | - | T | T | T | G | C | G | C | G | - | - | - | - | - | - | - | - | - | - | - | - | - |
| - | - | T | T | C | G | O | G | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |

In **Mixed Block Mode**, a block containing both gaps and data can be selected. If you touch on a cell containing data, then the smallest block bounded entirely by gaps (or the start or end of the matrix) is selected:

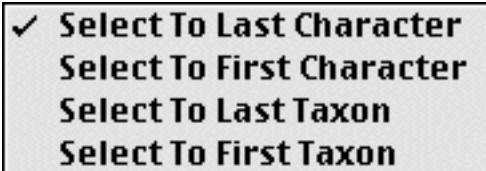


If a gap is selected with **Mixed Block Mode**, then the smallest block bounded entirely by nongaps (or the start or ends of the matrix) is selected.

If you want to select an entire sequence, not including any terminal gaps, then choose the **Select Entire Sequence** option in the Selection Wand's pop-up menu. This is available only if the tool is set to **Normal Mode**.

► **Select-to-end or Select-to-start tool**

This tool will select all of the cells from the cell touched to the start or end of the matrix. There are four modes, available from the tool's pop-up menu,

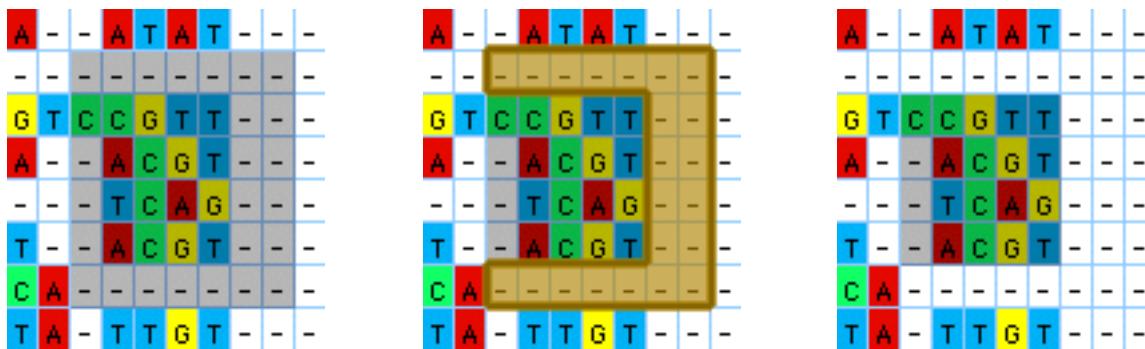


whose action is self-evident.

If you hold down the Option key, then MacClade will select in the reverse direction from that chosen in the pop-up menu.

Fitting the selection around data

If the boundaries of the selection contain all gaps on one side, then choosing **Shrink Wrap Selection** from the **Edit** menu will cause the selection to be shrunk by excluding from the selection any edges that are entirely gaps. For example, if **Shrink Wrap Selection** were applied to the selection shown in the leftmost figure below, then the part shown in brown (center) would be removed from the selection as those edges of the selection contain only gaps, reducing the selection to that shown on the right.



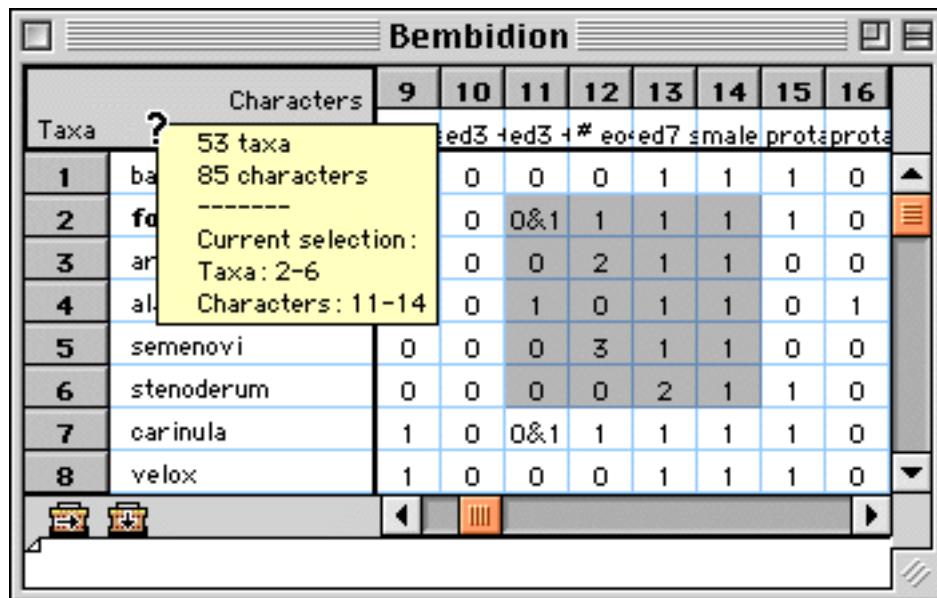
Shrink-wrapping a selection

This feature can be valuable when used with the molecular sequence editing tools in MacClade.

A cell containing missing is treated as a cell containing data (as opposed to treated like a gap) unless you have checked **Editor Tools Treat Missing as Gap** (see [page 210](#)).

Determining the location of the current selection

Touching on the rectangle in the upper left of the data editor will reveal the location of the current selection:



Undoing changes in the editor

The last change you make in the editor can be undone using the Undo item in the Edit menu, with a few exceptions. In particular, the following operations cannot be undone:

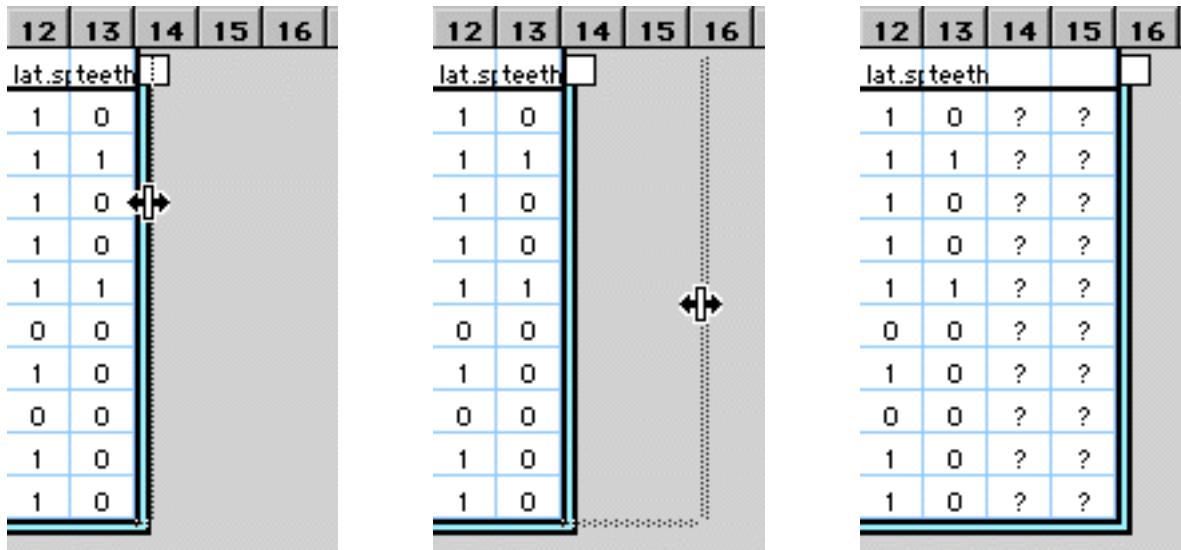
1. Deleting characters or taxa.
2. Filling character or taxon names.

Adding taxa or characters

This following sections describe the commands available that add, delete, or reorder whole taxa and characters. For most matrices a column will correspond to a character, and a row will correspond to a taxon, but the reverse will be true if you have transposed the matrix (["Choosing character by taxa or taxa by characters" on page 270](#)).

When you create a new MacClade data file, there is only one taxon and one character in the matrix. You will immediately need to add more taxa and, perhaps, more characters. You can do this by either grabbing and extending the matrix using the cursor, or by using **Add Empty Characters** in the **Characters** menu or **Add Empty Taxa** in the **Taxa** menu.

Add rows onto the edges of the matrix by grabbing along the lower matrix border and pulling downward. Add columns by grabbing along the right matrix border and pulling to the right. For example, pulling the edge of the matrix two columns to the right will add two columns to the matrix, as shown in the following sequence:



Adding columns by pulling on the matrix border

To insert one or more new columns between two existing columns, move the cursor to the line between the numbers at the top of those columns. The cursor will become a bar with an arrow on it. Then click, and drag to the right the required number of columns. When you release the mouse button, a space will open

up and new columns will be inserted. The following sequence shows addition of one character between characters 5 and 6:

| | 4 | 5 | 6 | 7 |
|--------|---|---|---|---|
| therrr | 0 | 0 | 0 | 0 |
| int. | 0 | 1 | 1 | 0 |
| ncat | 0 | 1 | 1 | 0 |
| atria | 0 | 1 | 1 | 0 |
| temp | 0 | 1 | 1 | 0 |
| | 0 | 1 | 1 | 2 |
| | 0 | 1 | 1 | 2 |
| | 1 | 1 | 1 | 2 |
| | 1 | 1 | 1 | 1 |
| | 0 | 1 | 1 | 2 |

| | 4 | 5 | 6 | 7 | 8 |
|--------|---|---|---|---|---|
| therrr | 0 | 0 | 0 | 0 | ? |
| int. | 0 | 1 | 1 | 0 | 0 |
| ncat | 0 | 1 | 1 | 0 | 1 |
| atria | 0 | 1 | 1 | 0 | 0 |
| temp | 0 | 1 | 1 | 0 | 0 |
| | 0 | 1 | 1 | 2 | ? |
| | 0 | 1 | 1 | 2 | 1 |
| | 1 | 1 | 1 | 2 | 0 |
| | 1 | 1 | 1 | 1 | 1 |
| | 0 | 1 | 1 | 2 | 2 |

| | 4 | 5 | 6 | 7 | 8 | ? |
|--------|---|---|---|---|---|---|
| therrr | 0 | 0 | 0 | 0 | 0 | 0 |
| int. | 0 | 1 | 1 | 0 | 1 | 0 |
| ncat | 0 | 1 | 1 | 0 | 1 | 0 |
| atria | 0 | 1 | 1 | 0 | 1 | 0 |
| temp | 0 | 1 | 1 | 0 | 1 | 0 |
| | 0 | 1 | 1 | 2 | ? | 1 |
| | 0 | 1 | 1 | 2 | 1 | 2 |
| | 1 | 1 | 1 | 2 | 1 | 2 |
| | 1 | 1 | 1 | 1 | 1 | 1 |
| | 0 | 1 | 1 | 2 | 1 | 2 |

Inserting a new column

Follow a similar procedure for inserting rows.

To add multiple characters quickly to the end of the matrix, use the **Add Empty Characters** dialog box from the **Characters** menu. The maximum number of characters allowed by MacClade 4.0 is 16,000. To add multiple taxa quickly to the end of the matrix, use the **Add Empty Taxa** dialog box from the **Taxa** menu. The maximum number of taxa allowed by MacClade 4.0 is 1,500.

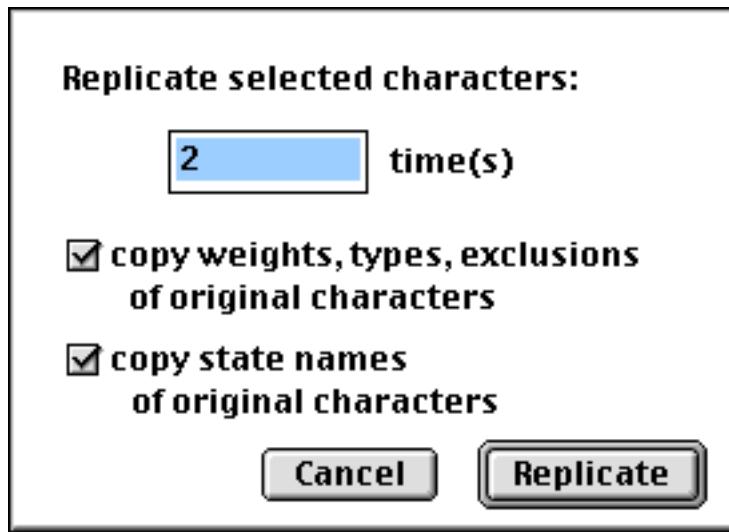
MacClade does not automatically open up new columns or rows to receive columns or rows that are being pasted from the Clipboard. See the section "[Copying and pasting](#)", page 235, for details.

Making duplicates of taxa or characters

If you wish to make a copy of one or more taxa or characters, you can select the rows or columns to be duplicated and use the **Duplicate** command in the **Edit** menu. If you have selected characters, then the types, weights, excluded/included status, and state names will be duplicated as well. Footnotes and pictures attached to cells will not be duplicated.

If you wish to make more than one copy or to have more control over the duplication, use the **Replicate** dialog box from the **Edit** menu. Select the rows or columns to be replicated. Choose **Replicate** from the

Edit menu. In the dialog box that appears enter the number of times you want the selected rows or columns to be replicated, and press **Replicate**:



For example, if two rows are selected:

| | | | | | | | |
|---|-------------|---|---|---|---|---|---|
| 4 | lungfish | 0 | 0 | 0 | 0 | 1 | 1 |
| 5 | salamanders | 0 | 1 | 1 | 0 | 1 | 1 |
| 6 | crocodiles | 1 | 1 | 2 | 0 | 1 | 1 |
| 7 | lizards | 1 | 1 | 2 | 0 | 1 | 1 |

and you ask to **Replicate** them two times, the result would be:

| | | | | | | | |
|----|---------------|---|---|---|---|---|---|
| 4 | lungfish | 0 | 0 | 0 | 0 | 1 | 1 |
| 5 | salamanders | 0 | 1 | 1 | 0 | 1 | 1 |
| 6 | crocodiles | 1 | 1 | 2 | 0 | 1 | 1 |
| 7 | salamanders.2 | 0 | 1 | 1 | 0 | 1 | 1 |
| 8 | salamanders.3 | 0 | 1 | 1 | 0 | 1 | 1 |
| 9 | crocodiles.2 | 1 | 1 | 2 | 0 | 1 | 1 |
| 10 | crocodiles.3 | 1 | 1 | 2 | 0 | 1 | 1 |
| 11 | lizards | 1 | 1 | 2 | 0 | 1 | 1 |

If you are multiplying characters, you can choose in the **Replicate** dialog box whether or not you want the weights, types, exclusion/inclusion, and state names of the original characters to be copied over to the new characters.

Taxa or characters can also be duplicated manually using **Copy** and **Paste** (see the section beginning on [page 235](#)), although some information (weights, types, exclusion/inclusion, and state names of the original characters) may be lost in this process.

Deleting taxa, characters, and cell blocks

MacClade has several means to remove taxa or characters from the data matrix. To remove taxa or characters by hand in the data matrix, select the *whole* row or column (see "[Data editor tools](#)" on page 207 for information on how to select a whole row or column), and choose **Cut** from the **Edit** menu (or press the Delete or Backspace key). To delete rows or columns from the ends of the matrix, grab the border of the matrix (see "[Adding taxa or characters](#)" on page 219), and push up or to the left to "squeeze" out the unwanted rows or columns. Taxa can also be deleted using the **Redundant Taxa** command (page 187), which will remove redundant taxa, or the **Merge Taxa** command (page 186), which will fuse several taxa into one. Characters can also be deleted using the **Compress Characters** command (page 200).

Before deleting taxa or characters from the data matrix, remember that MacClade allows exclusion of taxa from specific trees in the tree window, and allows temporary exclusion of characters in tree calculations. These exclusions are reversible, and may at times be more desirable than deleting forever the taxon or character from the data matrix. Deleting of taxa and characters is not reversible.

If whole rows or columns are not selected when you choose **Clear** or **Cut** or hit the Delete or Backspace key, then the taxa or characters will not be removed, but any data cells will be emptied (that is, filled with missing data). Note that you cannot clear blocks of taxon or character names in this manner.

Taxa that are deleted in the editor are automatically deleted from the tree in the tree window, from any stored trees in the data file, and from trees in any currently open tree file. States fixed at branches of a character traced in the tree window are lost if any taxa are deleted in the data editor.

Excising blocks using the scissors

The scissors excise pieces of the matrix. This tool removes the data and moves the remainder of those sequences to the left.

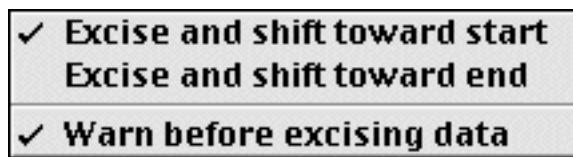
It can be applied to previously selected blocks, or it can be dragged to select the cells to be deleted. For example, if you used it on following selected block:

| | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| C | C | G | A | - | - | - | - | - | A | T | T | A | A |
| C | C | A | A | - | - | - | - | - | - | A | T | T | |
| C | C | A | A | - | - | - | - | - | - | A | T | T | |
| C | C | A | C | - | - | - | - | - | - | A | T | T | |
| C | C | G | A | - | - | - | - | - | A | T | T | A | A |

those 6 cells would be removed from the matrix and the remainder of those three sequences would be shifted to the left:

| | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| C | C | G | A | - | - | - | - | - | A | T | T | A | A |
| C | C | A | A | - | - | - | - | - | A | T | T | A | |
| C | C | A | A | - | - | - | - | - | A | T | T | A | |
| C | C | A | C | - | - | - | - | - | A | T | T | A | |
| C | C | G | A | - | - | - | - | - | A | T | T | A | A |

The scissor's pop-up menu



lets you change the mode of the tool. If you choose **Excise and shift toward end**, MacClade will shift the start of the sequence toward the end to fill the void:

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| C | C | G | A | - | - | - | - | - | A | T | T | A |
| A | G | C | C | A | A | - | - | - | - | A | T | T |
| A | G | C | C | A | A | - | - | - | - | A | T | T |
| A | G | C | C | A | C | - | - | - | - | A | T | T |
| C | C | G | A | - | - | - | - | - | A | T | T | A |

Switching between these modes can also be accomplished by holding down the Option key.

By default, MacClade warns you if the block to be excised contains any data; this warning can be turned off in the tool's pop-up menu.

Reordering taxa or characters

To move rows or columns, you first need to select the row(s) or column(s) to be moved. When you move the mouse over the numbers of the selected rows or columns, the cursor should change to a hand. You can then drag the rows or columns to the place you want them to be. For example, in the three images below, character 36 is being moved to be between characters 38 and 39.

| 35 | 36 | 37 | 38 | 39 |
|------|-------|-------|-------|-----|
| male | proto | flage | flage | CSC |
| 0 | 3 | 0 | 0 | 1 |
| 0 | 3 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 | 1 |
| 0&1 | 0 | 1 | 0 | 1 |
| 2 | 1 | 1 | 0 | 1 |
| 0 | 3 | 1 | 0 | 1 |
| 0 | 3 | 0 | 0 | 1 |
| 0 | 3 | 0 | 0 | 1 |
| 0&1 | 2 | 0 | 0 | 1 |
| 0 | 0 | 0 | 0 | 1 |

| 35 | 36 | 37 | 38 | 39 |
|------|-------|-------|-------|-----|
| male | proto | flage | flage | CSC |
| 0 | 3 | 0 | 0 | 1 |
| 0 | 3 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 | 1 |
| 0&1 | 0 | 1 | 0 | 1 |
| 2 | 1 | 1 | 0 | 1 |
| 0 | 3 | 1 | 0 | 1 |
| 0 | 3 | 0 | 0 | 1 |
| 0 | 3 | 0 | 0 | 1 |
| 0&1 | 2 | 0 | 0 | 1 |
| 0 | 0 | 0 | 0 | 1 |

| 35 | 36 | 37 | 38 | 39 |
|------|-------|-------|-------|-----|
| male | flage | flage | proto | CSC |
| 0 | 0 | 0 | 3 | 1 |
| 0 | 0 | 0 | 3 | 1 |
| 0 | 1 | 0 | 0 | 1 |
| 0&1 | 1 | 0 | 0 | 1 |
| 2 | 1 | 0 | 1 | 1 |
| 0 | 1 | 0 | 3 | 1 |
| 0 | 0 | 0 | 3 | 1 |
| 0 | 0 | 0 | 3 | 1 |
| 0&1 | 0 | 0 | 2 | 1 |
| 0 | 0 | 0 | 0 | 1 |



Moving a column by grabbing its number and dragging

Rows and columns can also be moved manually using **Cut** and **Paste**, but this is not recommended, as some critical information will be lost in the process (see ["Copying and pasting" on page 235](#)).

There are a number of other ways to reorder taxa and characters: they can be reordered in the list window (see "[Rearranging the order of objects](#)" on page 175) and using the sorting tool in the editor (see next section). In addition, taxa can be randomly reordered or moved to match the order of the taxa across the top of the current tree ("[Reordering taxa](#)" on page 185).



Sorting tool

This tool reorders the taxa or characters in either ascending or descending order. (You can toggle between ascending and descending either by using the tool's pop-up menu or by holding down the Option key.) Whether taxa or characters are sorted and how they are sorted depends upon the cell touched:

1. If you touch on a character name, then the characters will be sorted alphabetically by name.
2. If you touch on a taxon name, then the taxa will be sorted alphabetically by name.
3. If you touch on the number of a taxon, then the characters are sorted by their state in that taxon.
4. If you touch on the number of a character, then the taxa are sorted by their state in that character.

The fourth sorting mode is illustrated in the two figures shown below.

| | | Characters | | | |
|-----------|----------------|------------|-------|------|------|
| | | 1 | 2 | 3 | 4 |
| | | amni | upper | body | ther |
| 1 | rayfinned fish | 0 | 0 | 0 | 0 |
| 2 | frogs | 0 | 1 | 1 | 0 |
| 3 | turtles | 1 | 1 | 2 | 0 |
| 4 | lungfish | 0 | 0 | 0 | 0 |
| 5 | salamanders | 0 | 1 | 1 | 0 |
| 6 | crocodiles | 1 | 1 | 2 | 0 |
| 7 | lizards | 1 | 1 | 2 | 0 |
| 8 | birds | 1 | 2 | 3 | 1 |
| 9 | mammals | 1 | 1/2 | 4 | 1 |
| 10 | snakes | 1 | ? | 2 | 0 |
| 11 | | | | | |

| | | Characters | | | |
|-----------|----------------|------------|-------|------|------|
| | | 1 | 2 | 3 | 4 |
| | | amni | upper | body | ther |
| 1 | rayfinned fish | 0 | 0 | 0 | 0 |
| 2 | lungfish | 0 | 0 | 0 | 0 |
| 3 | frogs | 0 | 1 | 1 | 0 |
| 4 | salamanders | 0 | 1 | 1 | 0 |
| 5 | turtles | 1 | 1 | 2 | 0 |
| 6 | crocodiles | 1 | 1 | 2 | 0 |
| 7 | lizards | 1 | 1 | 2 | 0 |
| 8 | snakes | 1 | ? | 2 | 0 |
| 9 | birds | 1 | 2 | 3 | 1 |
| 10 | mammals | 1 | 1/2 | 4 | 1 |
| 11 | | | | | |

The ascending sort tool, applied to the top of character 3 (shaded in these figures for emphasis), as shown on the left, sorts the taxa so that those with state 0 are at the top, followed by those with state 1, 2, and so on, as shown in the right.

Entering data

Editing a cell

To enter data into a cell in the matrix, select the cell by clicking on it and type in the symbol or name of a state, as shown below. Editing the contents of a cell follows standard Macintosh text editing rules.

| Characters | | 1 | 2 | 3 | 4 | 5 | 6 |
|------------|---------|------|---|---|---|---|---|
| | | Taxa | | | | | |
| 1 | taxon 1 | 0 | X | ? | ? | ? | ? |
| 2 | taxon 2 | ? | ? | ? | ? | ? | ? |
| 3 | taxon 3 | ? | ? | ? | ? | ? | ? |
| 4 | taxon 4 | ? | ? | ? | ? | ? | ? |

Entering "0" for taxon 1,
character 1

(For ways to select and move between cells while entering data, see "[Selecting individual cells](#)" on [page 210](#) and "[Moving the selection from cell to cell](#)" on [page 211](#).)

Exactly what you type into the cell to indicate a character state depends upon the data file format (standard, extended standard, DNA, RNA, protein) and various other options you can choose. If you type into a cell and MacClade responds with "Bad entry in cell of data editor", you have made an illegal entry; see the section below ([page 229](#)) for explanation.

You can enter a state into a data cell in one of three ways:

1. If you have defined a name for states in the **State Names & Symbols** window (e.g., "red" or "blue"), and "interpret state names" is checked in the **Edit Interpretation** dialog box from the **Edit** menu, then you can type in the full name of the state. If you have not already named the states using the **State Names & Symbols** window, MacClade won't know what names like "red" or "blue" mean.
2. If you have defined symbols in the **State Names & Symbols** window (e.g., "a b c d"), then you can use one of those symbols. If there are predefined symbols (for DNA, RNA, or protein data), then you can use one of those predefined symbols (e.g., "A").
3. Finally, you can use one of the default symbols ("0 1 2 3 4 5 6 7 8 9" for standard data, "0 1 2 3 4 5 6 7 8 9 A B C D E F G H J K L M N P Q R" for extended standard data).

NOTE: To help you remember which symbols correspond to which named states, you can turn on **Footstates** in the **Display** menu — MacClade will then list the state symbols and names for the currently selected character in the footnote box at the bottom of the editor. See the section "[Footstates](#)" on [page 272](#) for an illustration.

When the "matching first taxon" option is turned on (see "[Matching first taxon](#)" on [page 258](#)), entering the matching character (e.g., ".") into a cell of the second or later taxon will cause MacClade to copy into that cell the same states found in the first taxon.

MacClade's interpretation of an entry

The following table shows an example in which the user has defined symbols "a" through "z" to be the symbols used for a data matrix (third column). See [page 198](#) for an explanation of state numbers. The first three columns apply to all characters in the data matrix. The numbers corresponding to the states are given in the first column, the default symbols in the second column, and symbols specified by the user in the third column. The fourth column, which lists state names that have been given to character 1, applies only to character 1. For example, if you wanted to enter state number 1 into a data cell for that character, then you could enter either "blue" or "b" or "1".

**Example of correspondence between
state numbers, symbols and state names**

| State number | Default symbol | User-defined symbol | State name of first character |
|--------------|----------------|---------------------|-------------------------------|
| 0 | 0 | a | aquamarine |
| 1 | 1 | b | blue |
| 2 | 2 | c | chartreuse |
| 3 | 3 | d | dark green |
| 4 | 4 | e | emerald |
| 5 | 5 | f | forest green |
| 6 | 6 | g | green |
| 7 | 7 | h | hot pink |
| 8 | 8 | i | indigo |
| 9 | 9 | j | jet black |
| 10 | A | k | khaki |
| 11 | B | l | lilac |
| 12 | C | m | magenta |
| 13 | D | n | nut brown |
| 14 | E | o | orange |
| 15 | F | p | purple |
| 16 | G | q | quail blue |
| 17 | H | r | red |
| 18 | J | s | sienna |
| 19 | K | t | tuscan red |
| 20 | L | u | umber |
| 21 | M | v | vermillion |
| 22 | N | w | white |
| 23 | P | x | xanthum |
| 24 | Q | y | yellow |
| 25 | R | z | zinc gray |

When you enter data for a cell, and if "interpret state names" is checked in the **Entry Interpretation** dialog box from the **Edit** menu, MacClade first checks to see if you have defined any state names for the character. If so, it checks for state names in the text you have typed. If these do not account for all the text you have typed, MacClade then checks to see if the symbols you have entered are present in your user-defined sym-

bols list (or predefined symbols list for molecular data). If it still has not found a full match, it then looks to see if there is a match between the entered symbols and the default symbols list.

For example, if you entered "9" into a cell of the character shown in the above table, MacClade would first check the state names in the fourth column, and discover that "9" was not a state name. It would then check the user-defined symbols list (third column), and find that it was not in the user-defined symbols list. Finally, it would check the default symbols list, and find that "9" corresponded to state number 9. Having found the state number is 9, MacClade would reassign to that taxon state number 9, which would be represented on the screen by symbol "j" or, if the character was character number 1 and **State Names In Cells** was turned on (["Displaying state names" on page 255](#)), by the state name "jet black".

For protein data, MacClade predefines as state names three-letter amino acid names (e.g., Ala, Leu). You can enter these state names instead of the IUPAC code if you wish.

For DNA or RNA data, MacClade first checks the predefined symbols list, then the default symbols list, as given in the following table.

| State number | Default symbol | Predefined symbols |
|--------------|----------------|--------------------|
| 0 | 0 | A |
| 1 | 1 | C |
| 2 | 2 | G |
| 3 | 3 | T |

NOTE: This feature can facilitate quick entry of some sorts of data. For example, if you have DNA sequence data, rather than typing in ACGT, the keys for which are scattered around the keyboard, you could type in 0123 on the numeric keypad. If you are entering a large amount of data, you may wish to place ACGT labels on top of the 0123 keys. Autotabbing (["Moving the selection from cell to cell" on page 211](#)) can also speed entry.

Terminal taxa of uncertain state or with polymorphism

MacClade allows a terminal taxon to be given more than one state in a character. There are two interpretations of a taxon coded with multiple states. The taxon's state might be uncertain, being one of the states listed, but exactly which one is not known. Second, the taxon may have been observed to have more than one state. This may occur if the terminal taxon is a species and shows a genetic polymorphism; it may occur if the terminal taxon is a collection of species that show various states and it is not known which is ancestral for the taxon. For either a species or higher taxon, we will speak of such a taxon with multiple observed states as being "polymorphic." The distinction between uncertainty and polymorphism for unorderd and ordered characters will not affect ancestral state reconstructions and comparisons between trees. However, uncertainties and polymorphisms may behave quite differently with irreversible, stratigraphic, Dollo, and user-defined type characters. MacClade does not treat taxa coded as having polymorphisms as if they have missing data, unlike the unnamed computer programs described by Nixon and Davis (1991). This distinction can be seen clearly in the different behavior of taxa coded with states 0&2 versus 1&2&3.

NOTE: MacClade allows a single data matrix to have some entries coded as polymorphisms, and other entries coded as uncertainties. Other programs, such as PAUP 3.0, require uniformity throughout a data matrix.

Before you fill your data matrix with polymorphisms, you should think carefully about the cautions given in the section ["Coding character states" on page 42](#).

To indicate in the editor that a taxon is polymorphic, type in the states separated by the AND separator (e.g., "0&1&2", meaning that the taxon has states 0 and 1 and 2). If the state in the taxon is entirely uncertain (i.e., it might be any state), then use the missing data symbol ("?"). If the state in the taxon is partially uncertain (for instance it isn't state 3, but it might be states 0, 1, or 2), then use the OR separator (e.g., "0/1/2", meaning that the taxon has states 0 or 1 or 2). The default AND separator, OR separator, missing data, and gap symbols are "&", "/", "?", and "-", respectively, but these can be changed using **State Names & Symbols** window available in the **Characters** menu.

NOTE: If you type both AND and OR separators into a cell (e.g., 0/1&2), MacClade will use the rightmost separator.

For DNA and RNA data sets, if you turn on **IUPAC Symbols in Cells** in the **Show** submenu of the **Display** menu, you can enter partial uncertainty using the standard IUPAC ambiguity symbols, if you wish. These symbols and their meanings are:

| Symbol | Meaning | Symbol | Meaning | Symbol | Meaning |
|---------------|----------------|---------------|----------------|---------------|----------------------|
| R | A or G | S | C or G | V | A or C or G |
| Y | C or T | W | A or T | D | A or G or T |
| M | A or C | H | A or C or T | X | A or C or G or T = ? |
| K | G or T | B | C or G or T | N | A or C or G or T = ? |

For example, you could enter "R", and MacClade would interpret this to mean "A or G".

If you type in more than one state in a cell, without typing in state separators (e.g., "012"), MacClade will by default assume you mean polymorphism (0&1&2). You can change this default to uncertainty using the **Entry Interpretation** dialog box in the **Edit** menu.

WARNING: If you enter states using full state names, and you enter more than one state in a cell, you must separate the states by either the AND separator or the OR separator. For example, entering "redblue" is not legal; "red&blue" is.

If you wish to change a number of polymorphisms to uncertainties, or the reverse, use the **Search & Replace** command from the **Utilities** menu (see "[Search & Replace](#)" on page 239).

Missing data and gaps

As noted above, if a taxon's state is completely uncertain, you should assign to it the missing data symbol ("?" as default). In a MacClade analysis, such a taxon is treated almost as if it didn't exist for this character's ancestral state reconstruction.

The **missing data** coding should be used with caution (Platnick et al., 1991; W. Maddison, 1993). W. Maddison (1993) points out that coding an organism as having missing data because the character does not apply to the organism (e.g., feather type in a crocodile) can lead to undesirable effects if such inapplicable codings are not isolated to a single group on the tree. When the character is inapplicable to a taxon, in general it would be better to code it as having an extra state ("not present"). The missing data coding would be better restricted to those circumstances in which the organism is presumed to have one of the character states, but which one it has is simply not known because of incomplete information. Nixon and Davis (1991) note that another use of the missing data coding, for polymorphisms, is also inappropriate. Nixon and Davis suggest that some computer programs that accept polymorphic codings for terminal taxa automatically treat them as if they had missing data; this is not the case with MacClade. These issues are discussed also in [Chapter 3](#).

The **gap** symbol is designed for molecular data, to indicate that there is a deletion (or an insertion in other sequences) such that the particular site does not exist in some organism; it can also be used equivalently

for nonmolecular data. Missing data and gaps are treated identically in MacClade's calculations with phylogenetic trees. The distinction is maintained by MacClade in the data matrices because it is useful for other programs or future versions of MacClade. Treating gaps like missing data can yield problems noted by Platnick et al. (1991) and W. Maddison (1993). One alternative available in PAUP*, treating gaps like an extra character state, can solve these problems but has the other problem of counting adjacent gaps as independent and thus inflating the cost of an insertion or deletion (Swofford, 2000).

"Bad entry in cell of data editor"

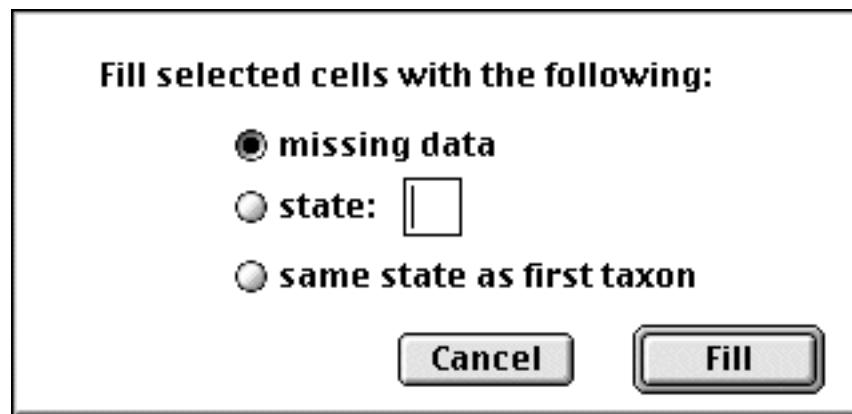
The bad-cell-entry message is given by the editor when (1) you have typed into a cell something other than a valid character state symbol or name, state separator, or missing data or gap symbol; or (2) you have typed into a cell a valid state name, but "interpret state names" is not checked in the **Entry Interpretation** dialog box in the **Edit** menu; or (3) you have typed into a cell a valid symbol or name to a character state that is not currently allowed because of the transformation type assigned to a character. The first problem might arise if you typed in nonsense characters, such as "[!]*+#}", or if you have typed in a name for a character state such as "small" without having first defined state names using **State Names & Symbols** in **Characters**. Remember, you have to define a state name like "small" before using it. The third problem might arise if you type in a "9" to a character currently assigned the type Dollo (in which 9 is not an allowable state), or a "2" to a character assigned a character state tree type defined only for states 0, 1, and 3. MacClade is conscious of the character transformation types assigned to the characters even though you are in the data editor. Generally, the default type assigned is unordered, which will allow any character states allowed by the data format. However, if any of your characters are of user-defined or Dollo types, you should be careful in the editor; these types may not allow a character to have any state. To check what character transformation types are assigned to the characters, choose **Character List** in **Characters**. For more information about character transformation types, see [Chapter 4](#).

If you do encounter the bad-cell-entry message, you will be presented with a choice: either discard the bad cell entry, and carry on as if you never typed the bad entry, or maintain the improper entry, but go back and select the cell with the improper entry, so that you may fix it.

Filling a block with data

There are several methods available in MacClade to fill a block of cells quickly, either using the **Fill** dialog box or some of the editor's tools.

The **Fill** dialog box from the **Utilities** menu allows you to specify the states to place in each cell of a selected region, if the region selected includes data cells:



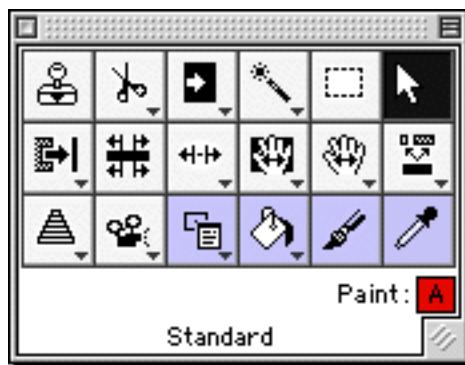
You can ask MacClade to fill the cells with missing data, with a single state, or with the same state as the

first taxon. Use state symbols to specify the state, not state names. If you type in more than one state (say "012"), then it will fill them only with the first state listed ("0"). You can fill a block of cells with gaps by entering the gap symbol in the box beside "state".

You can also fill a block of data cells with missing data by selecting them and choosing **Clear** from the **Edit** menu.

To fill a block of selected cells with random data you can use **Random Fill** in the **Utilities** menu; see "[Randomly assigning states to data cells](#)" on page 433 for further details.

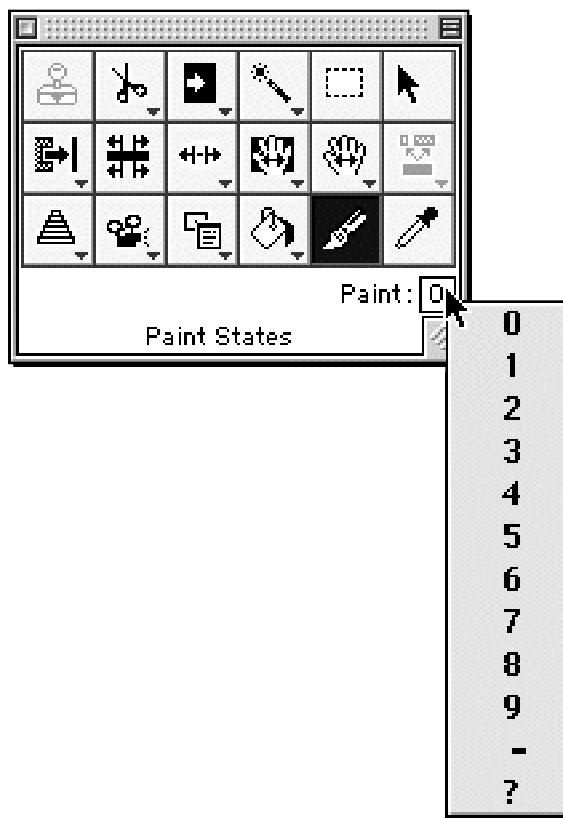
There are four tools in the data editor used for filling states of cells as described in the following sections. They are shaded in the figure below:



Setting the editor's "paint"

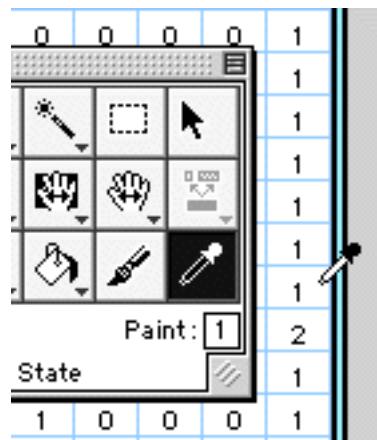
The paint states tool and the fill states tool both can fill cells with the state represented by the "paint". The state represented by the paint is indicated in the lower right-hand corner of the tool palette, and can be set either by using the choose state tool (page 231) or touching on the rectangle indicating the paint's state. If you touch on the box indicating the state, a menu will pop up allowing you to choose the state of the

paint:



Choose state (Eyedropper) tool

This tool allows you to choose the state used by the paint states and fill states tools. If you touch this tool on a cell in the matrix, it will pick up the state of that cell. For example, if you touched the tool on a cell containing state 1, then the state of the "paint" used by the paint states tool and the fill states tool would be 1, as indicated in the lower right-hand corner of the tool palette:



If the cell contains multiple states (as a polymorphism or uncertainty) then the paint will be similarly configured.

If you touch on a taxon name or a taxon number, then the paint state will be whatever the state is in that

taxon for each character. For example, if you touched on the number for taxon 3, then the paint would be whatever taxon 3's state is. In the example shown below, taxon 3's states in the first 11 characters are 11010001150:

The screenshot shows a character matrix with 11 columns and 5 rows of taxa. The taxa are labeled 1 through 5, and the characters are labeled 1 through 11. The matrix contains binary data (0 or 1). Row 3 (taxon 3) has values: 1, 0, 1, 0, 3, 3, 1, 0, 0, 3, 0. A cursor is pointing at the value '3' in the 11th column of row 3. Below the matrix is a 'Paint' tool dialog box. The 'Paint' dropdown menu is set to 'Taxon 3'. The 'Choose State' dropdown menu is open, showing a list of states: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 0*, 1*, 2*, 3*, 4*, 5*, 6*, 7*, 8*, 9*. The state '3' is highlighted.

The paint states tool or fill states tool would then fill any character 1 cell with 1, character 2 cell with 1, character 3 cell with 0, and so on:

The screenshot shows the same character matrix and Paint tool dialog as the previous image. However, the 'Paint' dropdown menu is now set to 'Fill States'. The 'Choose State' dropdown menu is also open, showing the same list of states as before. The state '3' is still highlighted. The matrix values remain the same as in the previous image, except for the 11th column of row 3, which is now filled with the value '3' (the state of taxon 3).

Paint states tool

This tool "paints" any cell it touches with the state represented by the paint (["Setting the editor's "paint"](#) on page 230).

Fill states tool



Use this tool on a selected block of cells to fill those cells with either the state represented by the paint ("Setting the editor's "paint"" on page 230), or with random data. This tool will also fill an unselected block of cells if you drag the tool over an unselected block.

There are several modes of action of this tool, selectable in the tool's pop-up menu:



The modes are:

1. **Fill with Paint:** The normal mode, in which the cells are filled with the paint.
2. **Fill with Random [Set Frequencies]:** Cells are filled with random data, in the frequencies set in the **Fill Random** dialog box in the **Utilities** menu (see "[Randomly assigning states to data cells](#)" on page 433).
3. **Fill with Random [Observed States, Equal Frequencies]:** Cells are filled with only the states *already present* in each character, in equal frequencies. Thus, if you used that on character 1 in the matrix below, with states 0 and 1, then the cells would be filled with 0 or 1, each with a probability of 0.5; if you used it on character 2, with states 0,1, and 3, then the cells of that character would be filled with states 0,1, or 3, each with a probability of 0.3333.

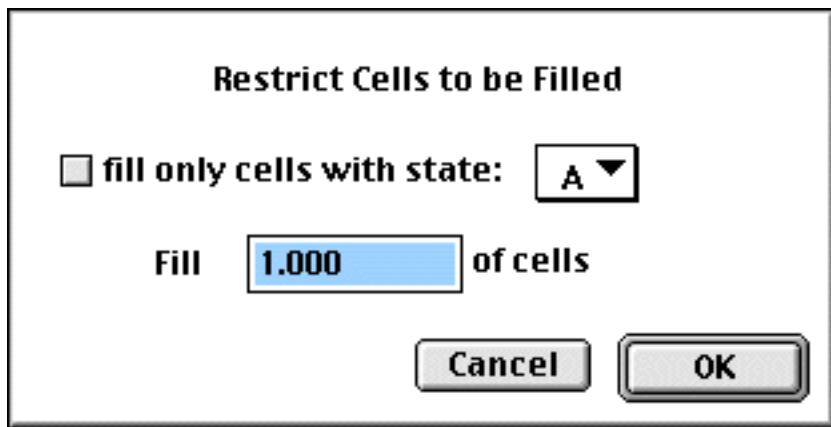
| 1 | 2 |
|---|---|
| 0 | 0 |
| 0 | 0 |
| 1 | 0 |
| 1 | 0 |
| 1 | 0 |
| 1 | 0 |
| 1 | 1 |
| 1 | 1 |
| 1 | 3 |
| 1 | 3 |
| ? | ? |
| ? | ? |

4. **Fill with Random [Observed States, Observed Frequencies]:** Cells are filled with only the states *already present* in each character, with probabilities proportional to the observed frequencies. Thus,

if you used that on character 1 above, which had two taxa with state 0 and 8 with state 1, then the cells would be filled with 0 with a probability of 0.2, and 1 with a probability of 0.8.

Filling only some of the selected cells

The **Restrict Fill** option (available in the pop-up menu) lets you restrict the fill, by presenting you with a dialog box containing options:



If you ask it to fill only cells with state X, then MacClade will ignore all other cells in the selected area, and only fill those with state X. Choosing "not (?) or -" from the pop-up menu in the dialog box will tell MacClade to only fill cells containing data.

You can also specify what proportion of the cells to fill. If you choose 0.1, then MacClade will fill each of the chosen cells with a probability of 0.1; thus, on average, only 10% of the cells will be altered. For example, if you asked MacClade to fill only cells with missing data, and asked it to fill 0.5 of the cells with the observed frequencies, then MacClade would fill 50% of the missing data with states present in each character, and fill them in the observed frequencies. This would allow you to investigate some effects of missing data. You could also investigate the effects of errors on your data by, say, filling 0.001 of the cells containing data with random data. You might also wish to explore filling with missing data a fraction of the cells.

If a fill is restricted in this way, then the tool will show a lock symbol on it: .

Pop-up data entry tool

Touch this tool on a cell or a selected block of cells, and a menu will pop up listing the states for that character. By selecting a menu item, that state will be entered into those cells.

For example, using the pop-up data entry tool on a block of cells:

| | | | | | |
|---|---|---|---|---|---|
| ? | ? | ? | ? | ? | ? |
| ? | ? | ? | ? | ? | ? |
| ? | ? | ? | ? | ? | ? |
| ? | ? | ? | ? | ? | ? |
| ? | ? | ? | 0 | ? | ? |
| ? | ? | ? | 1 | ? | ? |
| ? | ? | ? | 2 | ? | ? |

will fill those cells with the chosen state:

| | | | | | |
|---|---|---|---|---|---|
| ? | ? | ? | ? | ? | ? |
| ? | 1 | 1 | 1 | 1 | ? |
| ? | 1 | 1 | 1 | 1 | ? |
| ? | 1 | 1 | 1 | 1 | ? |
| ? | 1 | 1 | 1 | 1 | ? |
| ? | 1 | 1 | 1 | 1 | ? |
| ? | ? | ? | ? | ? | ? |

This tool has another mode, in which it shows the cell contents; this mode is available in the tool's pop-up menu. This can be handy if the contents of the cell are not entirely visible, perhaps as the column width is too small to display the entire cell's contents:

| | |
|-----------------------------------|---|
| act mis&disjoinedb 3rd&diamet | 2 |
| act mis&disjoinedo on diamet | 2 |
| act misct n distinct mirrors | 2 |
| act misct mid&iso 3rd&diamet+ spa | |

Copying and pasting

To copy or cut a portion of the matrix, and put it into the Clipboard, select the block (see "[Selecting rows, columns, and blocks](#)" on page 212), and choose **Copy** or **Cut** from the **Edit** menu. Remember that **Copy** just places a copy of the selected block in the Clipboard; **Cut** both places a copy into the Clipboard and gets rid of the original selection. Whether **Cut** deletes whole taxa and characters or merely clears the entries in the block's cells depends on whether whole taxa or characters were selected (see [page 212](#)).

| 2 | 3 | 4 | 5 | 6 | 7 |
|-----|-------|------|------|------|--------|
| i | upper | body | ther | int. | natria |
| 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 0 | 1 | 1 | 0 |
| 1 | 2 | 0 | 1 | 1 | 0 |
| 0 | 0 | 0 | 1 | 1 | 0 |
| 1 | 1 | 0 | 1 | 1 | 0 |
| 1 | 2 | 0 | 1 | 1 | 2 |
| 1 | 2 | 0 | 1 | 1 | 2 |
| 2 | 3 | 1 | 1 | 1 | 2 |
| 1/2 | 4 | 1 | 1 | 1 | 1 |
| ? | 2 | 0 | 1 | 1 | 2 |

A selected block of cells,
ready to be cut

| 2 | 3 | 4 | 5 | 6 | 7 |
|-----|-------|------|------|------|--------|
| i | upper | body | ther | int. | natria |
| 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 0 | 1 | 1 | 0 |
| 1 | ? | ? | ? | ? | 0 |
| 0 | ? | ? | ? | ? | 0 |
| 1 | ? | ? | ? | ? | 0 |
| 1 | ? | ? | ? | ? | 2 |
| 1 | ? | ? | ? | ? | 2 |
| 2 | ? | ? | ? | ? | 2 |
| 1/2 | 4 | 1 | 1 | 1 | 1 |
| ? | 2 | 0 | 1 | 1 | 2 |

Cells after cutting

To paste, select the region into which you wish to paste, and choose **Paste**. If MacClade thinks that you are trying to paste into a region different in size (different numbers of taxa and characters) from the Clipboard, it beeps. If you attempt to paste a clipboard of entire rows or columns containing more rows or columns than the current selection, MacClade will automatically insert extra rows or columns to accommodate the pasted elements.

Moving blocks of data can be accomplished either with **Cut** and **Paste**, or using the various alignment tools ([page 240](#)).

WARNING: Note that **Copy** or **Cut** will place only the character state data themselves into the Clipboard; the character state, taxon names, character weights, types, footnotes, and pictures in any column or row cut or copied will not be placed in the Clipboard. If you wish to rearrange characters, it would be best to use the technique for moving columns and rows (described on [page 223](#)) rather than **Copy** and **Paste**, as moving does retain all character information.

WARNING: It is best not to use **Cut** and **Paste** to move a taxon from one part of the matrix to another. If you were to select a taxon *Bigidae*, choose **Cut**, then go to another part of the matrix, insert a row, then select it and choose **Paste**, you would find that the taxon's states would be pasted into the row. If you then give this new taxon the name *Bigidae*, expecting that you have simply moved it, you will be mistaken. When you move to the tree window you will find that the *Bigidae* has been deleted from your trees. The reason is that each terminal taxon in MacClade's trees is associated with a particular row (or column) of the matrix; it is not associated with the taxon's name. A row's (or column's) integrity and association with trees is maintained if its taxa are reordered, or if its taxon is renamed, or if other taxa are inserted into the matrix, but the row (or column) vanishes when it is cut. The new row (or column), even though it has the same taxon name, is treated as a different taxon. To move taxa, it would be best to use the technique for moving columns and rows (described on [page 223](#)) rather than **Copy** and **Paste**.

If you copy data into the Clipboard, then switch data formats, the Clipboard is converted to the new data format when you ask MacClade to paste the data into the matrix. However, if nucleotide sequence data are in the Clipboard, and then you switch to protein data, the matrix itself will be *not* be translated if you ask to paste the data. If the data in the Clipboard contain more states than allowed under the current data format (e.g., the Clipboard data are of extended standard format, and the current file is of standard format) then MacClade will remove all unallowable states, and replace them by missing data.

WARNING: MacClade will paste only into a pre-existing cell, column, row, or other block of cells of the appropriate size. When you paste columns or rows, for instance, you must first have the correct number of columns or rows ready and selected to receive those pasted. You must make sure not only that the space is available, but also that a block of the exact same size is selected before pasting.

Stamp clipboard tool

This tool allows you to more easily use the clipboard to paste in blocks of cells. Select a block of cells, and copy it to the clipboard. Then choose this tool, and you will see and be able to move the clipboard contents around the editor. To paste the contents onto a region, click the mouse button to stamp the cells down.

| | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|-----------|-------|-------|-------|------|-------|-------|---|---|
| upperbody | therm | int.n | atria | temp | hemis | suspe | | |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ? |
| 1 | 1 | 0 | 4 | 1 | 0 | 0 | 0 | 0 |
| 1 | ? | ? | 2 | 0 | 1 | 1 | 0 | 0 |
| 0 | ? | ? | 0 | 0 | 1 | 1 | ? | 0 |
| 1 | ? | ? | 1 | 0 | 1 | 1 | 0 | 0 |
| 1 | ? | ? | 2 | 0 | 1 | 1 | 0 | 0 |
| 1 | ? | ? | 2 | 0 | 1 | 1 | 1 | 0 |
| 2 | ? | ? | 3 | 1 | 1 | 1 | 0 | 0 |
| 1/2 | 4 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| ? | 2 | 0 | 1 | 1 | 2 | 1 | 1 | 0 |

Pasting using the stamp tool

Copying data from one file to another

MacClade 4 does not have fully developed features for using the Clipboard to copy data from one MacClade file to another, or from MacClade to, for instance, a word-processing program. If you select and copy a block of a data matrix, then move to another MacClade data matrix, you can paste the block into the other matrix (as long as you have space in the other matrix, and paste the block as described above), but all assigned names, weights, and types will *not* be copied. Only the underlying data are copied. If you want to make a modified version of a data file, it would be better to duplicate the file (using **Save As** in MacClade or **Duplicate** in the Finder), then cut unneeded portions out of the duplicate version.

Data matrices cannot be copied from MacClade into another program using the Clipboard. Also, you cannot copy a matrix from another program and paste it into the MacClade data editor using the Clipboard. See [Chapter 8](#) to learn how you can move data matrices to and from other programs.

Recoding character data

MacClade provides two features that allow you to search and replace, *en masse*, entries in the data matrix: **Recode**, and **Search & Replace**. The simpler tool is the **Recode** dialog box (**Utilities** menu), which allows

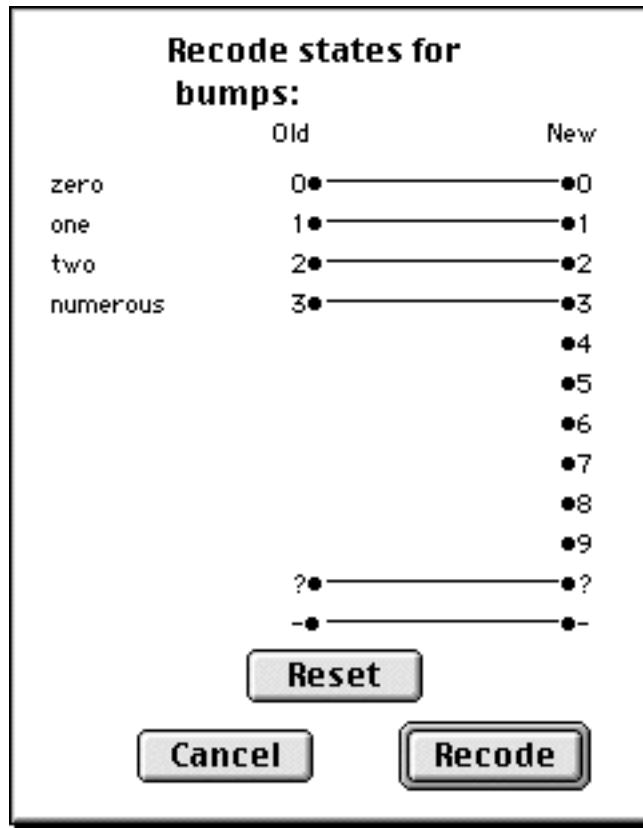
you to recode many states of the selected character or characters at one time (e.g., change all 0 to 1, all 1 to 0, and all 2 to 1 in the selected characters). The **Search & Replace** dialog box (Utilities menu) is more flexible, in that it can also change combinations of states (e.g., change all 0&1 to 2/3), but it can't simultaneously recode many states at one time.

(For recoding DNA/RNA sequence data as protein data, see the section on "[Data formats](#)" on page 192. For producing the complement of DNA/RNA sequence data, see [Chapter 16](#).)

Recoding by either **Recode** or **Search & Replace** changes the data for each taxon and character affected. They differ from changing the Symbols list in two ways: they actually change the state numbers stored by MacClade, rather than changing just the symbols by which the states are referred (see [page 198](#) for discussions of numbers, symbols, and names), and they apply only to the characters and taxa requested. **Search & Replace** differs from **Recode** in that the former changes only the underlying state numbers; the latter also readjusts the state names so that the recoded cells display the same state names.

The Recode dialog box

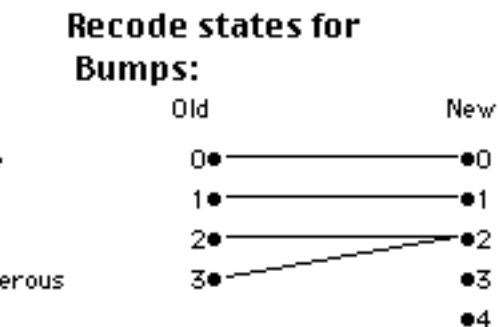
If one or more whole characters are selected, you can use the **Recode** dialog box to recode their states.



In the dialog box are two columns indicating state symbols, the left column for the old coding of characters and the right for the new. On the leftmost part of the dialog box are the names of the states, which are shown only if the character has names and only one character is being recoded.

Grab the dot corresponding to the old coding of a state and drop it on the dot for the new coding of the state. When you press Recode, all taxa with this old state will be recoded as having the new state (for the selected character(s) only). For example, in the illustration below, the user has decided that instead of

dividing the character into four states, "none", "one", "two", and "numerous", she or he prefers the coding "none", "one", and "more than one".



This will cause states 2 and 3 to be fused by making all instances of state 3 recoded to be state 2. After recoding, the user will notice that the state names will be listed as "none", "one", "two-numerous", because MacClade fuses the names of any fused states during recoding. The user can then go to the **State Names & Symbols** window to change "two-numerous" to "more than one", and the new coding has been achieved.

Whole character(s) must be selected in the data editor before recoding. If more than one character is selected when **Recode** is called, all of the characters selected will be recoded similarly.

Recode redefines the correspondence between states and their names, so that a taxon will keep the same state name in the character even if its underlying state number and symbol have changed. For this reason **Recode** can be readily used to try alternative orderings of character states. Suppose you have a character but can't decide whether its states are ordered red, blue then green, or red, green then blue. Using State Names, indicate state 0 is red, 1 is blue, 2 is green. Enter the data as red, green, or blue. Try this character as ordered, then compare the results to a recoding in which 1 is recoded for 2 and 2 for 1. This would test the alternative ordering red-green-blue; the taxa would still be listed as having states red, green, or blue as you entered.

In the example illustrated above, recoding was used to group together states formerly thought distinct. Though this may be valid if evidence now points to the states actually being the same, recoding is not always the best way to group states together. Suppose, for example, that your species either show no chromosome fusions, or any of several different chromosome fusions. If you wanted to trace chromosome evolution to see which ancestors might have had chromosome fusions (without regard to just what fusions they had), you might recode the character so that the unfused condition is state 0 and all fused states become state 1. This would, however, allow two different fusions to be placed next to one another on the tree with no required character steps, contrary to what is known about the chromosomes. A better approach would be to avoid recoding; leave all the information in the character, but reassign the patterns or colors used by MacClade's character tracing so that all of the fused chromosome character states receive the same pattern or color (as explained under "[Changing patterns or shades or colors](#) on page 355). In this way, the reconstruction uses the character's full information, but the shading of patterns in the character tracing summarizes the results as unfused versus fused.

Search & Replace

The **Search & Replace** dialog box (Utilities menu) allows you more flexibility than **Recode**, but is restricted to making one kind of change at a time. (The **Find** command in MacClade is separate from **Search & Replace**, and is described in "[Finding sequences of states](#) on page 252.)

Enter into the "Search for:" box the states you want MacClade to search for, and enter into the "Replace with:" box those states you want them changed to. (You cannot use state names; you must use state symbols in this dialog box.)

If you choose "Selection only", then any changes will be made only to cells that are currently selected; if it is not checked, then changes are made throughout the matrix.

If "Whole cell" is selected, then the only cells that will be changed are ones that match exactly the state set listed in the "Search for:" box. For example, if you had "0" in the "Search for:" box, and "2" in the "Replace with:" box, then MacClade would change cells that contain "0&1" to "2&1" if "Whole cell" is *not* selected, and would not change such cells if it were selected.

Note that changing from a symbol to an equivalent symbol doesn't accomplish anything. For example, if symbols are defined such that the first state is "-" and the second is "+", then as "-" is the symbol for state number 0, changing "-" to 0 doesn't change the data.

You can also use this dialog box to change all uncertainties to polymorphisms or vice versa by entering the AND and OR separators in the "Search for:" and "Replace with:" boxes. For example, if you entered "/" into the "Search for:" box, and "&" into the "Replace with:" box, then uncertainties would be changed to polymorphisms. If you do change separators in this way, then the "Whole cell" option will be ignored.

Complementing nucleotide sequence data

If you select a block of cells and choose **Complement** from the **Utilities** menu, the bases in the selected block of cells will be replaced by their complements (that is, A will replace T or U, T or U will replace A, C will replace G, and G will replace C). This works only for nucleotide data, and is illustrated in [Chapter 16](#).

Reversing or inverting molecular sequence data

If you select a block of cells and choose **Reverse** from the **Utilities** menu, the block of cells will be reversed (that is, the entries for the first and last sites in the selected block will be switched, as will the second and second-to-last, and so on). This works only for DNA, RNA, or protein sequences, and is illustrated in [Chapter 16](#).

Changing gaps and missing data

To automatically change missing to gaps, or the reverse, or N's to missing, or the reverse, or terminal partial triplets to gaps, use the **Change All** submenu of the **Utilities** menu. Details of these commands are given under ["Changing gaps and missing data" on page 303](#).

Translating nucleotide sequence data

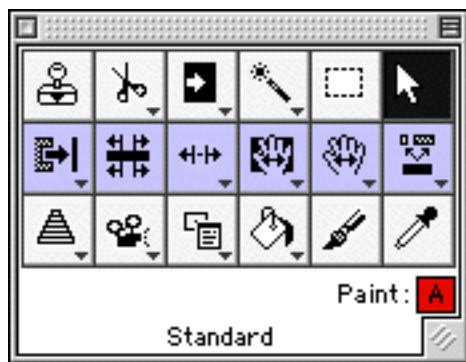
To translate nucleotide sequence data to the corresponding amino acids, choose **Translate to Protein** from the **Data Format** submenu of the **Characters** menu. This is described in detail in [Chapter 16](#).

Shuffling data

You can reshuffle existing data using the **Shuffle** command; see ["Shuffling existing data" on page 435](#) for details.

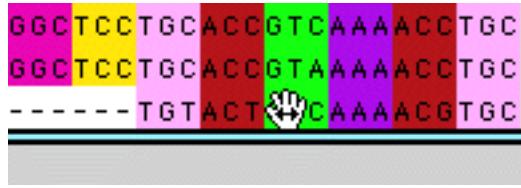
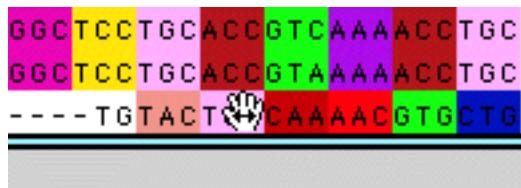
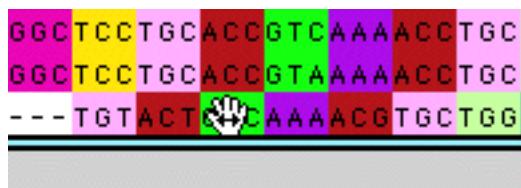
Moving data cells & sequence alignment

Six of the tools in MacClade's data editor tool palette are designed to aid in manual alignment of molecular sequences. They are shown shaded in the figure below:



The section ["Alignment of molecular sequences within MacClade"](#) on page 297 gives general information and advice about the use of these tools; the following sections provide details about available options and behavior of the tools.

With each of these tools, the screen display is updated automatically as you move the sequences. For example, if you use the Block Mover with nucleotide data colored by the amino acid to which they would be translated (["Coloring cells" on page 259](#)), then these colors are updated as you shift the reading frame of the sequence by moving it, as shown in the following sequence:



If consensus sequences are shown (["Consensus sequences" on page 295](#)) then they are automatically updated as you move a sequence, as shown in the following two images:



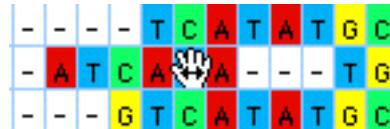
| Modal | | C T G C A C C G T H A A |
|-------|----------|-------------------------|
| 1 | insect 1 | C T G C A C C G T C A A |
| 2 | insect 2 | C T G C A C C G T A A A |
| 3 | insect 3 | - - T G T A + T G T C A |

| Modal | | C T G C A C C G T C A A |
|-------|----------|-------------------------|
| 1 | insect 1 | C T G C A C C G T C A A |
| 2 | insect 2 | C T G C A C C G T A A A |
| 3 | insect 3 | - T G T A + T G T C A |

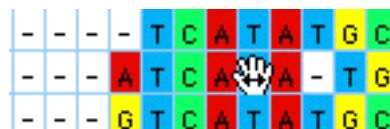
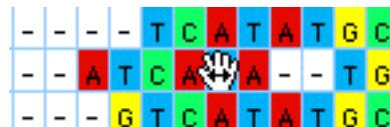
Block Mover

The Block Mover is the main manual alignment tool, allowing you to grab and move portions of a molecular sequence.

In default mode, this tool moves contiguous blocks of nucleotides or amino acids in single sequences. For example, if you grabbed the string of nucleotides ATCATA in the second sequence:



This tool would move those six nucleotides in the direction they are pushed:



The tool (in default mode) would not move any nucleotides outside of this block. A block consists of the largest string of cells that does not include gaps; if **Editor Tools Treat Missing as Gap** is selected in the Edit menu ([page 210](#)), then the string will not contain missing data. You will be able to move the block forward and backward through any bounding gaps; you will not be able to move it over any other nucleotides or amino acids. If you wish to move more than one sequence at once, you will need to use the Selected Block Mover ([page 243](#)) or the Split Others tool ([page 245](#)).

The pop-up menu for this tool,

Allow Matrix Edges to Expand
Move Entire Sequence
Accumulate Sticky Blocks
Allow Block Splitting

allows you to change its behavior. In particular:

1. If **Allow Matrix Edges to Expand** is checked, then MacClade will add more characters to the start or end of the matrix to make room for sequences pushed off the start or end.
2. If **Move Entire Sequence** is checked, then the entire sequence, exclusive of terminal gaps, will be moved, not just a single block of nucleotides or amino acids.
3. If **Accumulate Sticky Blocks** is checked, then as you move one piece along, if it bumps into another piece, then that other piece will be joined to the first piece and you will now be moving both pieces, as one.
4. If **Allow Block Splitting** is checked, then the Block Mover behaves as the Block Splitter (["Block Splitter" on page 244](#)) when the tool is over the boundary between two cells.

Selected Block Mover

This tool allows you to move selected blocks of data. It is somewhat similar to the Block Mover, but it only works on selected blocks. In contrast to the Block Mover, the Selected Block Mover does not choose the boundaries of the block to be moved; the block to be moved is whatever you have selected. It also allows you to move multiple sequences at once, if these have been selected. It is valuable in conjunction with the Selection Wand ([page 213](#)).

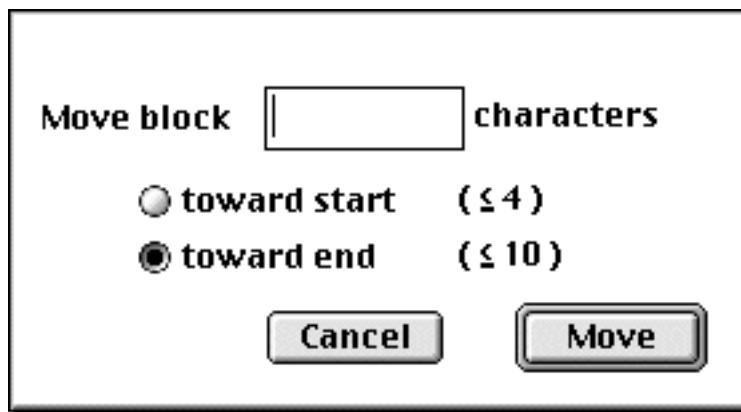
For example, using this tool on the block selected in the following image

| | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| - | C | T | G | C | A | C | G | T | C | A | A | A | A | C | C | T |
| - | - | - | C | T | G | C | A | G | G | T | A | A | A | A | A | C |
| - | - | - | T | G | T | A | C | I | G | T | C | A | A | A | A | C |

allows you to move that portion of the two sequences to the left:

| | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| - | C | T | G | C | A | C | G | T | C | A | A | A | A | C | C | T |
| - | C | T | G | C | A | C | G | T | A | A | A | A | A | - | - | C |
| - | - | T | G | T | A | C | I | G | T | C | A | A | A | - | - | C |

There are only two modes for this tool. In normal mode, you drag the selected block to its new position. In contrast, if **Shift Specified Amount** is chosen in the tool's pop-up menu, then when you click on the selected block, you will be presented with a dialog box

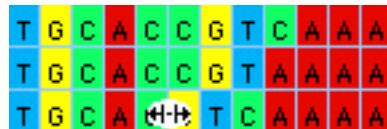


in which you can enter the number of positions towards the start or end you wish to shift the block. The maximum allowed shift will be noted in parentheses in the dialog box.

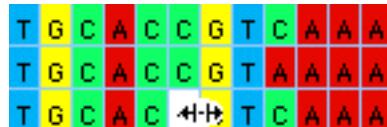
You can drag this tool through unselected regions to select them.

↔↔ Block Splitter

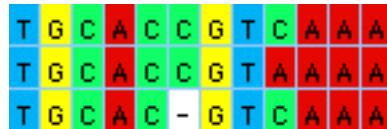
This tool allows you to split a block of nucleotides or amino acids in two, thereby introducing gaps between the two segments. For example, using the tool between two nucleotides in the last sequence:



allows you to move part of the last sequence to the right,



thereby introducing a gap:



The mode of action of this tool can be modified by choosing items in its pop-up menu.

If **Multiple Sequence Selection Mode** is chosen, then you can split multiple sequences at once, but first you must select the line of the split by dragging the tool down through the sequences on a line between two sites. A shimmering bar will appear; release the mouse button, and position the tool over the shim-

mering bar:



| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| T | G | C | A | C | C | G | T | C | A | A | A |
| T | G | C | A | C | C | G | T | A | A | A | A |
| T | G | C | A | C | G | T | C | A | A | A | A |
| T | G | C | A | C | G | T | C | A | A | A | A |
| T | G | C | A | C | G | T | C | A | A | A | A |
| T | G | C | A | C | G | T | C | A | A | A | A |
| T | G | C | A | C | G | T | C | A | A | A | A |
| T | G | C | A | C | G | T | C | A | A | A | A |

Dragging the shimmering bar to the left or right will allow you to split all of those sequences along that line:

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| T | G | C | A | C | C | G | T | C | A | A | A |
| T | G | C | A | C | C | G | T | A | A | A | A |
| T | G | C | A | C | - | G | T | C | A | A | A |
| T | G | C | A | C | - | G | T | C | A | A | A |
| T | G | C | A | C | - | G | T | C | A | A | A |
| T | G | C | A | C | - | G | T | C | A | A | A |
| T | G | C | A | C | - | G | T | C | A | A | A |
| T | G | C | A | C | - | G | T | C | A | A | A |

Holding the Option key down will toggle the tool between single sequence mode and multiple sequence mode.

The Block Splitter has two other options:

1. If **Move All Blocks on Side** is chosen, then the tool will move not just the portion of the block of nucleotides or amino acids being split, but the entire sequence or set of sequences on the one side, including gaps.
2. If **Allow Matrix Edges to Expand** is checked, then MacClade will add more characters to the start or end of the matrix to make room for sequences pushed off the start or end.

Split Others tool

This tool is similar to the Block Splitter except that it moves all sequences **other than** the one being dragged. For example, if you positioned the tool as shown:

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|--|--|
| G | A | T | C | G | G | C | G | A | | | |
| A | A | T | C | G | G | C | G | A | | | |
| G | A | T | A | | | G | C | G | A | | |
| C | A | T | C | G | G | T | G | A | | | |
| C | A | T | C | G | G | T | G | A | | | |

and pulled one column to the right, all other sequences would be moved to the right, and gaps inserted:

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| G | A | T | C | - | - | G | G | C | G | A |
| A | A | T | C | - | - | G | G | C | G | A |
| G | A | T | A | G | G | G | A | C | A | |
| C | A | T | C | - | - | G | G | T | G | A |
| C | A | T | C | - | - | G | G | T | G | A |

This tool moves the entire length of the sequences on the right or left sides, not just the adjacent block of data.

➡ Close Gaps tool

This tool moves a selected block toward the start or end as far as it can go through any existing gaps, pushing it up against the next non-gap. For example, if it were used on the block selected below:

| | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| C | C | G | T | A | A | A | A | - | A | - | - | - | - | C |
| C | G | T | C | - | A | A | A | - | A | - | - | - | - | C |
| C | G | T | C | - | A | A | - | A | - | - | - | - | - | C |
| C | G | T | C | - | - | A | A | A | - | - | - | - | - | C |
| C | G | T | C | - | - | A | A | A | - | - | - | - | - | C |
| C | G | T | C | - | - | A | A | A | - | - | - | - | - | C |

then the selected block would be moved to the right as far as possible:

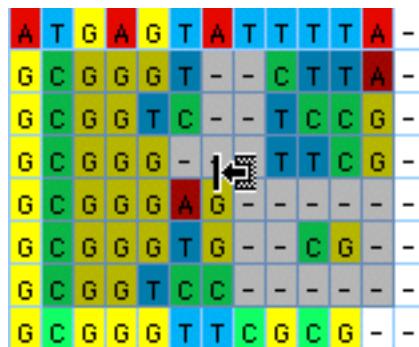
| | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| C | C | G | T | A | A | A | A | - | A | - | - | - | - | C |
| C | - | - | - | - | G | T | C | - | A | A | A | - | A | C |
| C | - | - | - | - | G | T | C | - | A | A | A | - | A | C |
| C | - | - | - | - | G | T | C | - | - | A | A | A | A | C |
| C | - | - | - | - | G | T | C | - | - | A | A | - | A | C |
| C | G | T | C | - | A | A | A | - | A | - | - | - | - | C |

By default, the tool moves the block toward the end of the sequences; if you wish to move the block toward the start of the sequences, then choose **Move as Far Toward Start as Possible** in the tool's pop-up menu, or hold down the Option key.

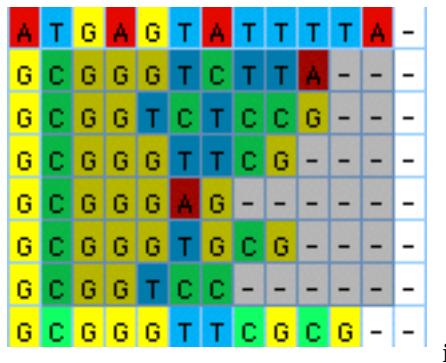
If you choose **Collapse All Contained Gaps** in the pop-up menu, then all gaps in the selected block will be removed, compressing the nucleotides or amino acids as far toward the beginning or end as possible:

| | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| C | C | G | T | A | A | A | A | - | A | - | - | - | - | C |
| C | - | - | - | - | - | G | T | C | A | A | A | A | - | C |
| C | - | - | - | - | - | G | T | C | A | A | A | A | - | C |
| C | - | - | - | - | - | G | T | C | A | A | A | A | - | C |
| C | - | - | - | - | - | G | T | C | A | A | A | A | - | C |
| C | G | T | C | - | A | A | A | - | A | - | - | - | - | C |

Collapsing contained gaps will work even if the block as a whole cannot be moved. For example, if **Collapse All Contained Gaps** is chosen and you request to move the nucleotides toward the start in the following selected block:



the block as a whole cannot be moved, as it is already abutting nucleotides, but the contained gaps can be collapsed:



i

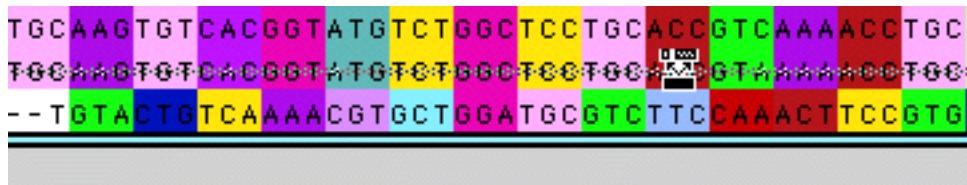
Pairwise alignment tool

This tool performs an alignment of two molecular sequences, one against another. It only does pairwise alignment; it does not perform multiple sequence alignments as do programs such as Clustal W (Thompson et al., 1994). It has a number of options and modes.

In the default mode, you drag one sequence onto another, and the tool aligns the dragged sequence against the one on which it is dropped (the receiving sequence), as shown in the following series:



Dragging the last sequence



Dropping it onto the second-last sequence (the receiving sequence)



MacClade aligns the last sequence agains the second-last

MacClade will insert gaps as needed into the sequences. For example, if you drag the second sequence onto the first using the pairwise alignment tool in the figure on the left, below, MacClade will insert a gap in the dragged sequence:

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | C | T | C | A | A | A | G | A | T | T | A | A |
| 2 | T | C | T | C | A | A | A | G | A | T | A | A |
| 3 | C | T | C | A | A | A | A | G | A | T | T | A |

Aligning sequence 2 against 1

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| C | T | C | A | A | A | A | G | A | T | T | A | A |
| C | T | C | A | A | A | A | G | A | T | - | A | A |
| C | T | C | A | A | A | A | G | A | T | T | A | A |

MacClade inserts a gap in sequence 2

The gaps might instead by inserted into the receiving sequence, if that is more parsimonious. If a gap is added to the receiving sequence, then gaps are added to all remaining sequences other than the one dragged:

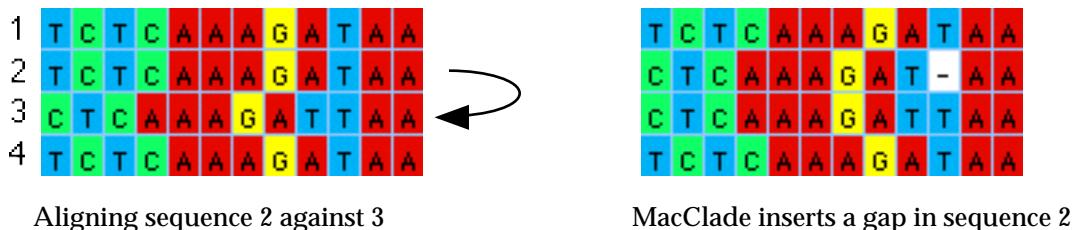
| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | T | C | T | C | A | A | A | A | G | A | T | A |
| 2 | T | C | T | C | A | A | A | A | G | A | T | A |
| 3 | C | T | C | A | A | A | A | G | A | T | T | A |
| 4 | T | C | T | C | A | A | A | A | G | A | T | A |

Aligning sequence 3 against 2

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| T | C | T | C | A | A | A | A | G | A | T | - | A |
| T | C | T | C | A | A | A | A | G | A | T | - | A |
| T | C | T | C | A | A | A | A | G | A | T | T | A |
| T | C | T | C | A | A | A | A | G | A | T | - | A |

MacClade inserts a gap in sequences 1-2, 4

Aligning sequence A against sequence B will not always yield the same results as aligning sequence B against sequence A. In the example immediately above, if you dropped sequence 2 onto sequence 3, the result would be insertion of a gap into sequence 2:



The cost of an alignment

MacClade finds the alignment between the two sequences that is most parsimonious; that is, the alignment for which the total costs of substitutions and gaps are minimal. To calculate the cost of a particular alignment between two sequences, the costs of various manipulations to convert one sequence into the other must be specified. For example, if you specify that changing the nucleotides or amino acids between one sequence and the next entails a cost of 2 (this is called the substitution cost), then the following alignment will have a cost of 4, as the two sequences differ only by two nucleotides:

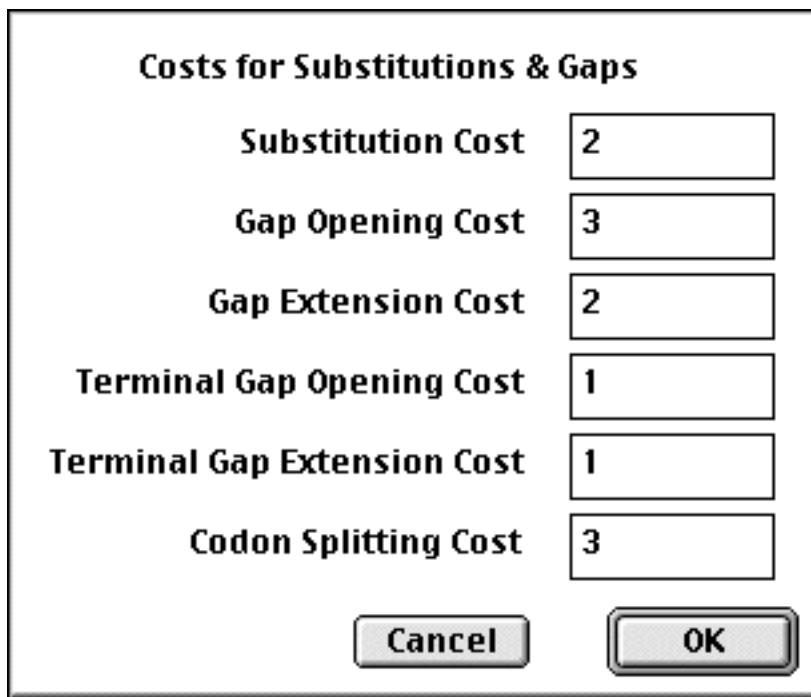


There is also a cost specified for creating a new series of gaps in a sequence (the "gap opening cost") and a cost for lengthening a series of gaps (the "gap extension cost"). If we call the gap opening cost P and the gap extension cost E, then the cost of a series of gaps of length G would be equal to $P + EG$. For example, if the gap opening cost were 3 and the gap extension cost 2, then the alignment on the left would have a cost of 5, and the alignment on the right a cost of 9:



If there are both substitutions and gaps that differ between the sequences, then the sum of their costs is used to calculate a total cost for the alignment. Different alignments will entail different costs, and MacClade will find the alignment with the lowest total cost.

In the addition to the substitution cost, the gap opening cost, and the gap extension cost, you can specify other costs in the Substitution and Gap Costs dialog box:

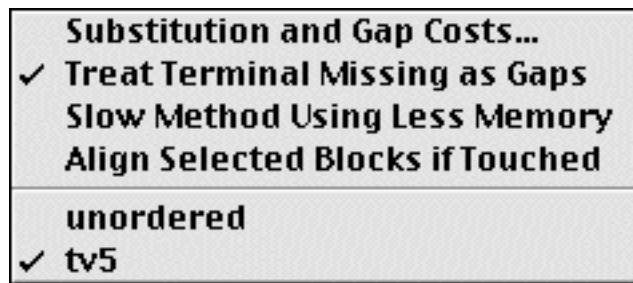


You can use a different gap opening and extension cost at the ends of the sequences than within them (these are the terminal costs shown above). In addition, you can specify an additional cost for introducing a gap in the receiving sequence if that gap splits a codon; for example, if a new gap is introduced between codon position 1 and 2, or 2 and 3. This cost is not invoked for opening a gap between positions 3 and 1. Of course, this cost is only relevant for nucleotide data in which codon positions have been specified for the receiving sequence (see ["Specifying coding regions and codon positions" on page 292](#)).

By default, all substitutions are given equal weight. If you wish to specify a different cost for one type of substitution relative to another (for example, you might wish to specify transversions as 5 times the cost of transitions), then you will need to create a user-defined type (["Defining and editing types" on page 285](#)). Once defined, user-defined step matrices will appear in the bottom of the pairwise alignment tool's pop-up menu, and you can then choose the type. For example, the user-defined type tv5 might be defined as follows:

| To: | A | C | G | T |
|---------|---|---|---|---|
| From: A | 0 | 5 | 1 | 5 |
| C | 5 | 0 | 5 | 1 |
| G | 1 | 5 | 0 | 5 |
| T | 5 | 1 | 5 | 0 |

This user-defined step matrix can then be selected in the pairwise alignment tool's menu:



When so chosen, MacClade will multiply the steps in the chosen step matrix by the substitution cost to calculate the cost of any particular substitution between two aligned sequences.

There is no substitution cost added if a missing data cell in one sequence is lined up against a nucleotide in the other. By default, terminal missing data cells (those found before the first specified nucleotide in a sequence or after the last specified nucleotide) are treated as gaps; if you wish them treated as missing data, de-select **Treat Terminal Missing as Gaps** from the tool's pop-up menu. Note that by default other missing data cells are treated as such; however, if you choose **Editor Tools Treat Missing as Gap** in the **Edit** menu, then such missing data cells will be treated as gaps during the alignment.

In calculating the cost of an alignment, MacClade respects the gaps already present in the receiving sequence. For example, if there are already some gaps in a region of the receiving sequence, then extending that region of gaps will entail gap extension costs, but no gap opening cost. If there is already a gap in the receiving sequence, then no cost will be incurred for opening up a gap of the same or smaller size within the same region.

In contrast, the gaps in the dragged sequence are entirely ignored by MacClade – they are thrown away before MacClade performs the alignment (although they might be added back in by the alignment tool).

Fast, memory-intensive alignment versus slow, memory-light alignment

MacClade can use two methods to conduct the alignment, which differ in the amount of memory they use and their speed. The default method uses a tremendous amount of memory, but it is relatively fast. If you find that MacClade complains, and states that there is not enough memory to conduct the alignment, then either you can increase the amount of memory given to MacClade (see "[Computer memory and MacClade's limits](#)" on page 112), or you can switch to the slower method that uses less memory by selecting **Slow Method Using Less Memory** from the pairwise alignment tool's pop-up menu. The memory used by the fast method is approximately three times as much as by the slow method; some examples of memory required and time taken (on a 400MHz G3 iMac) are shown in the following table:

| Number of nucleotides in sequence dragged | Length of receiving sequence | Memory required with fast method | Time taken with fast method | Memory required with slow method | Time taken with slow method |
|---|------------------------------|----------------------------------|-----------------------------|----------------------------------|-----------------------------|
| 200 | 200 | 487 Kb | 0.3 s | 163 Kb | 6.2 s |
| 500 | 500 | 3 Mb | 1.6 s | 1 Mb | 1 min 35 s |
| 1000 | 1000 | 12 Mb | 6.6 s | 4 Mb | 13 min 14 s |
| 2000 | 2000 | 48 Mb | 26 s | 16 Mb | 1 hr 48 min |
| 5000 | 5000 | 300 Mb | (not calc.) | 100 Mb | (not calc.) |

Automatic pairwise alignment of multiple sequences

By default, the pairwise alignment tool works by dragging one sequence upon another, with the sequence dragged aligned to the entire receiving sequence. In contrast, if you choose **Align selected blocks if touched** from the alignment tool's pop-up menu, then the tool will work only on a selected set of sequences or a selected block. If more than one sequence is present in the selected block, then MacClade will align all remaining sequences against the sequence touched. If only part of some sequences are selected, then MacClade will align only the portions selected.

For example, in this mode, if characters 5 through 12 were selected for four sequences, and you touched on the first sequence (left, below), then MacClade would first align sequence two against sequence one, then three against one, then four against one. Each of these would be a simple pairwise alignment. The regions of the sequences not selected would remain untouched (right, below).

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|-----|---|
| - | A | C | C | T | A | G | G | T | ACT | T |
| A | - | C | C | T | T | A | C | T | - | - |
| - | A | C | C | T | - | - | G | T | A | C |
| - | A | C | C | T | G | T | A | C | T | - |

Touching the pairwise alignment tool (in selected block mode) on the first sequence in a selection

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|-----|---|
| - | A | C | C | T | A | G | G | T | ACT | T |
| A | - | C | C | - | - | - | T | T | A | C |
| - | A | C | C | - | - | T | G | T | A | C |
| - | A | C | C | - | - | T | G | T | A | C |

The resulting alignment

If, on the other hand, the third sequence were touched, then this would serve as the receiving sequence, and each of the other sequences would be aligned to it:

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|-----|---|
| - | A | C | C | T | A | G | G | T | ACT | T |
| A | - | C | C | T | T | A | C | T | - | - |
| - | A | C | C | T | - | - | G | ▲ | A | C |
| - | A | C | C | T | G | T | A | C | T | - |

Touching the pairwise alignment tool (in selected block mode) on the third sequence in a selection

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|-----|---|
| - | A | C | C | T | A | G | G | T | ACT | T |
| A | - | C | C | T | - | - | - | T | A | C |
| - | A | C | C | T | - | - | G | T | A | C |
| - | A | C | C | T | - | - | G | T | A | C |

The resulting alignment

The pairwise alignment algorithms

The algorithm used by the pairwise alignment tool in its slow, less memory-intensive mode is similar to that described by Needleman and Wunsch (1970), modified to deal with the more flexible costs allowed by MacClade. The algorithm used by the pairwise alignment tool in its fast, more memory-intensive mode is similar to that described by Gotoh (1982), again modified to deal with the more flexible costs allowed by MacClade.

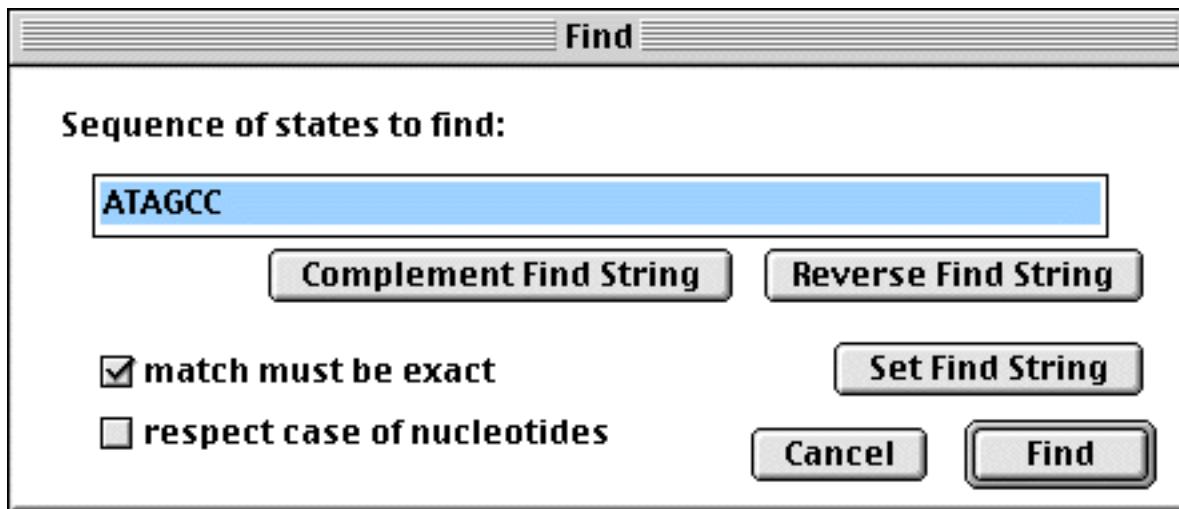
Aligning nucleotides to match a protein alignment

MacClade has a preliminary feature that allows you to realign nucleotide sequences to match the amino acid alignment in a protein sequence; see "[Aligning nucleotides to match a protein alignment](#)" on page 299.

Finding sequences of states

Especially when examining molecular sequence data, it can be useful to search for sequences of states. For

instance, the sequence "ATAGCC" might be sought because of its importance for some function, or as an aid in sequence alignment. Such sequences can be found using the **Find** dialog box available in the **Find** submenu in the **Utilities** menu:



Enter the desired sequence, and then hit **Find** to find the next occurrence of this sequence in the matrix. By selecting **Find Again** in the **Find** submenu, the next occurrence will be selected. You can also select **Flash Find String** to ask MacClade to highlight all such sequences throughout the matrix.

A convenient way to enter the string to be sought is to select a piece of one sequence in the editor, then choose **Enter Find String** from the **Find** submenu. You can then choose **Find Next** or **Find Previous**, and thus avoid the dialog box.

If you uncheck "match must be exact", then the match need not be exact. If ATA?CC were the string to find, then it would match only ATA?CC if the match had to be exact, but would match ATAACC, ATACCC, ATAGCC, or ATATCC if the match could be inexact. That is, if matches are allowed to be inexact, then if one string is ambiguous (because of missing data or partial uncertainty), then a string that is consistent with it will be considered a match.

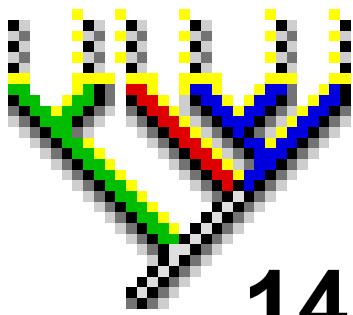
Finally, you can ask MacClade to flash all partial triplets in nucleotide protein-coding data by choosing **Flash Partial Triplets** from the **Find** submenu. A partial triplet is a set of three cells in the matrix within one sequence representing first, second, and third codon positions that contains only one or two nucleotides, not three as one would expect if there were not frame-shifts. For example, in the file below, the boxed region represents a partial triplet, as the three cells in the codon have TC-.

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|---|---|---|---|---|---|---|---|---|----|----|----|
| G | G | A | T | C | C | T | T | A | C | T | A |
| G | G | A | T | C | - | C | T | A | T | T | A |
| G | G | A | T | C | A | C | T | A | C | T | A |

Proofreading your matrix

You can ask MacClade to read your data matrix out loud if you have the appropriate software installed. To ask MacClade to speak your matrix, you need to be running System 7.1 or later, and to have the relevant speech synthesis software installed (Macintalk or PlainTalk; look for the Speech control panel).

Choosing **Speak Matrix** in the **Utilities** menu will then cause MacClade to read your matrix. If you have a single cell selected in the matrix, MacClade will begin speaking from that point; otherwise, it will begin at the start of the matrix. Speaking will continue until you hold the mouse button down for a while, or until you type command-period ($\text{⌘}.$). You can control the timing of pauses in reading the matrix, using the **Speech Options** dialog box in the **Utilities** menu. If you want MacClade to pause after every cell, enter "1" in the "every _ items" box.



DISPLAY OF DATA

14

MacClade has many options for altering the display of the data matrix. You can change various aspects of the display of the editor, including font of text and column widths; the matrix can also be transposed. The matrix can be drawn so as to place a simple symbol (e.g., ".") whenever a taxon has the same state as that in the first taxon. Footnotes and pictures can be attached to cells in the data matrix. Various special styles are available for sequence data, including coloring of cells as an alignment aid.

Display of data cells

Displaying state names

By default, MacClade displays the data cells in a matrix using state symbols:

| | | Characters | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | |
|---|----------------|------------|------|-------|------|------|------|----|-------|------|-------|-------|
| | | Taxa | amni | upper | body | ther | int. | ne | atria | temp | hemis | suspe |
| 1 | rayfinned fish | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ? | |
| 2 | frogs | | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | |
| 3 | turtles | | 1 | 1 | 2 | 0 | 1 | 1 | 0 | 0 | 0 | |
| 4 | lungfish | | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | ? | |
| 5 | salamanders | | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | |

To have the data editor display states by their full names, choose **State Names In Cells** from the **Show** submenu of the **Display** menu:

| Taxa | | Characters | | 1 amnion | 2 appendages | 3 body covering |
|------|----------------|------------|-----------|-------------|-----------------|--------------------|
| | | 1 | 2 | | | |
| 1 | rayfinned fish | absent | fins | derm.scales | | |
| 2 | frogs | absent | legs only | smooth | | |
| 3 | turtles | present | legs only | epid.scales | | |
| 4 | lungfish | absent | fins | derm.scales | | |
| 5 | salamanders | absent | legs only | smooth | | |

For full state names to appear in the editor, you must have predefined the state names in the **State Names & Symbols** window (see "[State names and symbols](#)" on page 194). There is an exception to this: Amino acid names are predefined for protein data sets. If you show data using full state names, you may wish to increase width of columns ([page 269](#)).

NOTE: *If you have chosen to display state names in the editor, you should also have "interpret state names" turned on in the **Entry Interpretation** dialog box in the **Edit** menu ([page 225](#)); otherwise, you may receive many bad-cell-entry messages. By default, "interpret state names" is turned on.*

Displaying states by their full names affects only the data editor display. In a saved NEXUS file on disk, MacClade writes the matrix using symbols and records the state names elsewhere in the file.

IUPAC symbols for DNA and RNA data

If you choose **IUPAC Symbols in Cells** from the **Show** submenu of the **Display** menu, MacClade will display DNA and RNA sequence data using the IUPAC symbols for partially uncertain data (see "[Terminal taxa of uncertain state or with polymorphism](#)" on page 227).

Three-letter amino acid names

By default MacClade displays amino acid names using the standard one-letter symbols. If you wish MacClade to display them using three-letter names, choose **Three-Letter AA Names** from the **Show** submenu of the **Display** menu. The three-letter names will then be used in display of protein data:

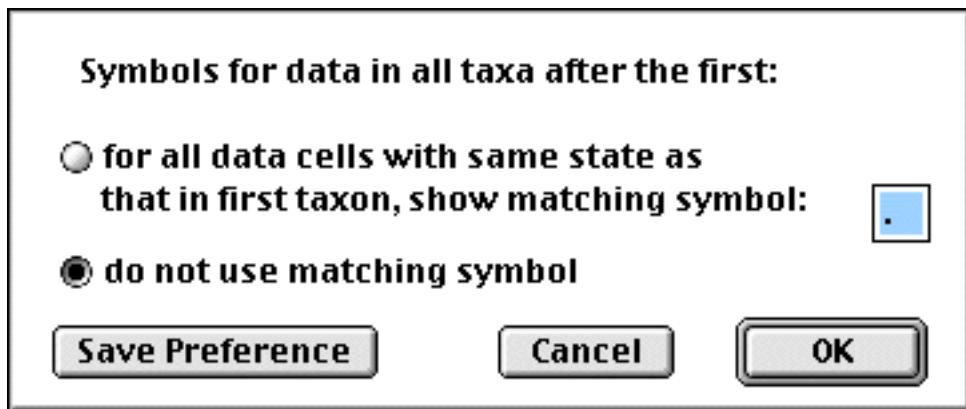
| Taxa | | Characters | 12 | 13 | 14 | 15 | 16 | 17 |
|------|--------------------------|------------|-----|-----|-----|-----|-----|-----|
| | | | | | | | | |
| 1 | Struthio camelus | | Glu | Val | Val | Leu | Pro | Lys |
| 2 | Rhea americana | | Glu | Val | Leu | Leu | Pro | Glu |
| 3 | Pterocnemia pennata | | Glu | Val | Leu | Leu | Pro | Glu |
| 4 | Casuarius casuarius | | Glu | Val | Leu | Leu | Pro | Lys |
| 5 | Dromaius novaehollandiae | | Glu | Val | Leu | Leu | Pro | Lys |
| 6 | Nothoprocta cinerascens | | Glu | Val | Pro | Leu | Pro | Lys |
| 7 | Eudromia elegans | | Glu | Val | Pro | Leu | Pro | Lys |
| 8 | Pygoscelis adeliae f | | Glu | Val | Leu | Val | Thr | Lys |
| 9 | Pygoscelis adeliae y | | Glu | Val | Leu | Val | Thr | Lys |
| | | T 11 C 18 | < | | | | | > |

and in the AA Translation mode for protein-coding nucleotide sequence (see "[Amino acid translation](#)" on [page 261](#)):

| | | | | | | | | | | | |
|-----|---|---|-----|---|---|-----|---|---|-----|---|---|
| A | G | C | T | T | T | A | C | A | G | G | T |
| Ser | | | Phe | | | Thr | | | Gly | | |
| A | G | C | T | T | T | T | C | C | G | G | C |
| Ser | | | Phe | | | Ser | | | Gly | | |
| A | G | C | T | T | T | T | C | T | G | G | C |
| Ser | | | Phe | | | Ser | | | Gly | | |

Matching first taxon

The data can be viewed in a format commonly used by molecular biologists with the **Match First** dialog box, available in the **Display** menu:



Choosing the upper option in the dialog box will make the first taxon the standard; for all other taxa, only those states that differ from that of the first taxon will be shown, otherwise the matching symbol (by default, a period ".") will be displayed:

| | | Characters | 2 | 4 | 2 | 5 | 2 | 6 | 2 | 7 | 2 | 8 | 2 | 9 | 3 | 0 | 3 | 1 | 3 | 2 | 3 | 3 |
|---|----------|------------|---|----------|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Taxa | 1 | insect 1 | G | C | G | A | C | T | T | C | C | A | | | | | | | | |
| 1 | insect 1 | | . | . | . | . | . | . | . | . | . | . | . | . | . | . | . | . | . | . | . | |
| 2 | insect 2 | | . | . | . | . | . | . | . | . | . | . | . | . | . | . | . | . | . | . | . | |
| 3 | insect 3 | | . | . | . | C | . | . | . | . | . | . | . | C | | | | | | | | |
| 4 | insect 4 | | . | . | . | T | . | . | C | . | . | . | . | C | . | . | . | . | . | . | . | |
| 5 | insect 5 | | . | . | . | C | . | . | C | . | . | . | G | | | | | | | | | |
| 6 | insect 6 | | . | . | . | . | . | . | C | . | . | . | . | C | . | . | . | . | . | . | . | |

By moving a different taxon into the first position, it will then become the standard. The matching symbol can be changed in the **Match First** dialog box. This option can be used for nonmolecular data or molecular data.

Coloring cells

The **Color Cells** submenu of the **Display** menu controls the colors used in the data cells in the editor. If "Don't Color" is selected (the default for nonmolecular data), then the cells are shown with the character states in black on white:

| | | Characters | 2 4 2 2 2 5 6 5 2 3 4 5 6 7 8 |
|---|----------|------------------|-------------------------------|
| | | Taxa | |
| 1 | insect 1 | GCGACTTCCAGCCTT | |
| 2 | insect 2 | GCGGTTACCAAGTTT | |
| 3 | insect 3 | GCGCCTTCCCAGTTT | |
| 4 | insect 4 | GCGTCTCCCAAGCTT | |
| 5 | insect 5 | GCGCCTCCCGAACTT | |
| 6 | insect 6 | GCGACTCCCAAACCTT | |

To have the cells of the matrix colored according to the character states in each cell, choose an item from the **Color Cells** submenu of the **Display** menu. For instance, coloring a nucleotide sequence matrix **By State** will yield the following display:

| | | Characters | 2 4 2 2 2 5 6 5 2 3 4 5 6 7 8 |
|---|----------|------------------|-------------------------------|
| | | Taxa | |
| 1 | insect 1 | GCGACTTCCAGCCTT | |
| 2 | insect 2 | GCGGTTACCAAGTTT | |
| 3 | insect 3 | GCGCCTTCCCAGTTT | |
| 4 | insect 4 | GCGTCTCCCAAGCTT | |
| 5 | insect 5 | GCGCCTCCCGAACTT | |
| 6 | insect 6 | GCGACTCCCAAACCTT | |

The colors used for the **By State** option can be changed by choosing **State Colors** in the **Color Cells** submenu.

For nucleotide protein-coding data, you may also ask to have the cells colored by the amino acid into which it would be translated, using the current designation of codon positions and the genetic code, by choosing **By Translated AA State** in the **Color Cells** submenu:

| | | Characters | 2 4 2 2 2 5 6 5 2 3 4 5 6 7 8 |
|---|----------|------------------|-------------------------------|
| | | Taxa | |
| 1 | insect 1 | GCGACTTCCAGCCTT | |
| 2 | insect 2 | GCGGTTACCAAGTTT | |
| 3 | insect 3 | GCGCCTTCCCAGTTT | |
| 4 | insect 4 | GCGTCTCCCAAGCTT | |
| 5 | insect 5 | GCGCCTCCCGAACTT | |
| 6 | insect 6 | GCGACTCCCAAACCTT | |

The colors used for the **By Translated AA State** option can be set by choosing **Amino Acid Colors** in the **Color Cells** submenu.

If you choose **Matching First Taxon** from the **Color Cells** menu, then all cells with the same state as the first taxon will be shaded in gray:

| | | Characters | 2 2 2 2 2 5 6 5 2 3 4 5 6 7 8 |
|---|----------|-------------------|-------------------------------|
| | | Taxa | |
| 1 | insect 1 | GCGACTTCCAGCCTT | |
| 2 | insect 2 | GCGGTTACCAAGTTT | |
| 3 | insect 3 | GCGCCTTCCCAGTTT | |
| 4 | insect 4 | GCGTCTCCCCAAGCTT | |
| 5 | insect 5 | GCGCCTCCCCGAACCTT | |
| 6 | insect 6 | GCGACTCCCCAAACTT | |

All of these options, except for **By Translated AA State**, are available for nonmolecular data. For example, if you color the cells in the "Vertebrates" example file by state, it will look as follows:

| | | Characters | 1 | 2 | 3 | 4 | 5 | 6 | 7 | |
|----|----------------|------------|------|-------|------|-------|------|-------|------|--|
| | | Taxa | amni | upper | body | therm | int. | atria | temp | |
| 1 | rayfinned fish | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| 2 | frogs | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | |
| 3 | turtles | 1 | 1 | 2 | 0 | 1 | 1 | 1 | 0 | |
| 4 | lungfish | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | |
| 5 | salamanders | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | |
| 6 | crocodiles | 1 | 1 | 2 | 0 | 1 | 1 | 1 | | |
| 7 | lizards | 1 | 1 | 2 | 0 | 1 | 1 | 1 | | |
| 8 | birds | 1 | | 3 | 1 | 1 | 1 | | | |
| 9 | mammals | 1 | 1/2 | | 1 | 1 | 1 | 1 | | |
| 10 | snakes | 1 | ? | 2 | 0 | 1 | 1 | | | |

By default, the colors used in each character are independent, and are determined by the number of states in the character (for example, a character with states 0 and 1 will use yellow and blue as the colors). If you

wish that all characters use the same colors, choose "use same colors for all characters" in the **State Colors** item in the **Color Cells** submenu. The "Vertebrates" matrix will then look as follows:

| | | Characters | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | |
|----|----------------|------------|-----|------|-----------|-------|------|--------|------|---|---|
| | | Taxa | | amni | upperbody | therm | int. | natria | temp | | |
| 1 | rayfinned fish | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ▲ |
| 2 | frogs | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | ☰ |
| 3 | turtles | 1 | 1 | 2 | 0 | 1 | 1 | 0 | 0 | 0 | |
| 4 | lungfish | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | |
| 5 | salamanders | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | |
| 6 | crocodiles | 1 | 1 | 2 | 0 | 1 | 1 | 1 | 2 | 0 | |
| 7 | lizards | 1 | 1 | 2 | 0 | 1 | 1 | 1 | 2 | 0 | |
| 8 | birds | 1 | 2 | 3 | 1 | 1 | 1 | 1 | 2 | 0 | |
| 9 | mammals | 1 | 1/2 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | |
| 10 | snakes | 1 | ? | 2 | 0 | 1 | 1 | 1 | 2 | 0 | ▼ |
| | | T 10 C 4 | | | | | | | | | |

Here, all characters use the same color pattern, that of the character having the highest numbered state.

Amino acid translation

MacClade has two ways to translate nucleotide sequences to amino acid sequences: first, by changing data formats of the matrix (selecting **Translate to Protein** in the **Data Format** submenu of the **Characters** menu), and second, by the simultaneous translation available in the **Display** menu. The first (described in "[Translating nucleotide sequences to amino acid sequences" on page 307](#)) alters the data themselves and is irreversible; the second merely displays translated amino acid sequences alongside the nucleotide sequence. These options are available only if you have designated codon positions ("[Specifying coding regions and codon positions" on page 292](#)).

To turn on the display of simultaneous amino acid translation, the items in the **Amino Acid Translation** submenu of the **Display** menu may be selected, or one of two special **Data Matrix Styles** can be chosen (**Nucleotide with AA Colors** and **Nucleotide with AA Translation**; see next section).

Choosing **Show Translated AA's** in the **Amino Acid Translation** submenu will add an extra line to each taxon showing the amino acid into which each codon would be translated. The nucleotides are automatically dimmed so that the amino acids are more obvious:

| Characters | | 1 | 1 | 2 | 1 | 3 | 1 | 4 | 1 | 5 | 1 | 6 | 1 | 7 | 1 | 8 | 1 | 9 | 2 | 0 | 2 | 1 | 2 | 2 | 2 | 3 | 2 | 4 | 2 | 5 | 2 | 6 | 2 | 7 | 2 | 8 | | | | | | | | |
|------------|--------------|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|--|--|--|--|--|
| Taxa | | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | | | | | |
| 1 | Homo sapiens | G | G | C | G | C | A | G | T | C | A | T | T | C | T | C | A | T | A | | | | | | | | | | | | | | | | | | | | | | | | | |
| | | G | | A | | V | | I | | L | | M | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 2 | Pan | G | G | C | G | C | A | A | T | T | A | T | T | C | C | C | T | C | A | T | A | | | | | | | | | | | | | | | | | | | | | | | |
| | | G | | A | | I | | I | | L | | M | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 3 | Gorilla | G | G | C | G | C | A | G | T | T | G | T | T | C | T | T | C | T | T | A | T | A | | | | | | | | | | | | | | | | | | | | | | |
| | | G | | A | | V | | V | | L | | M | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 4 | Pongo | G | G | C | G | C | A | A | C | C | A | C | C | C | C | C | T | C | A | T | G | | | | | | | | | | | | | | | | | | | | | | | |
| | | G | | A | | T | | T | | L | | M | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

If you do not wish the nucleotides to be dimmed, you can return them to their normal color by unchecking the **Dim Nucleotides** item in the **Amino Acid Translation** submenu:

| Characters | | 1 | 1 | 2 | 1 | 3 | 1 | 4 | 1 | 5 | 1 | 6 | 1 | 7 | 1 | 8 | 1 | 9 | 2 | 0 | 2 | 1 | 2 | 2 | 2 | 3 | 2 | 4 | 2 | 5 | 2 | 6 | 2 | 7 | 2 | 8 | | | | | | | | | | |
|------------|--------------|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|--|--|--|--|--|--|--|
| Taxa | | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | | | | | | | |
| 1 | Homo sapiens | G | G | C | G | C | A | G | T | C | A | T | T | C | T | C | A | T | A | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | | G | | A | | V | | I | | L | | M | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 2 | Pan | G | G | C | G | C | A | A | T | T | A | T | T | C | C | C | T | C | A | T | A | | | | | | | | | | | | | | | | | | | | | | | | | |
| | | G | | A | | I | | I | | L | | M | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 3 | Gorilla | G | G | C | G | C | A | G | T | T | G | T | T | C | T | T | C | T | T | A | T | A | | | | | | | | | | | | | | | | | | | | | | | | |
| | | G | | A | | V | | V | | L | | M | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 4 | Pongo | G | G | C | G | C | A | A | C | C | A | C | C | C | C | C | T | C | A | T | G | | | | | | | | | | | | | | | | | | | | | | | | | |
| | | G | | A | | T | | T | | L | | M | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

If you choose **Color Translated AA's** from the **Amino Acid Translation** submenu, the amino acids will be colored:

| Characters | | 1 | 1 | 2 | 1 | 3 | 1 | 4 | 1 | 5 | 1 | 6 | 1 | 7 | 1 | 8 | 1 | 9 | 2 | 0 | 2 | 1 | 2 | 2 | 2 | 3 | 2 | 4 | 2 | 5 | 2 | 6 | 2 | 7 | 2 | 8 | | | | | | | | | | | | |
|------------|--------------|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|--|--|--|--|--|--|--|--|--|
| Taxa | | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | | | | | | | | | |
| 1 | Homo sapiens | G | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | | G | | A | | V | | I | | L | | M | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 2 | Pan | G | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | | G | | A | | I | | I | | L | | M | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 3 | Gorilla | G | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | | G | | A | | V | | V | | L | | M | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 4 | Pongo | G | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | | G | | A | | T | | T | | L | | M | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

Data matrix styles

With the many options for display of data matrices it may take several steps to set the display to that desired. For convenience MacClade has a **Data Matrix Styles** submenu of the Display menu, which automatically chooses several options at once in order to set the display to a commonly used form.

For example, the **Plain** style, available for all data formats, will set the matrix to use a standard column width, have uncolored cells, and so on:

| Characters | | 27 | 28 | 29 | 30 | 31 | 32 | 33 | 34 | 35 | 36 | 37 | 38 | 39 | 40 | 41 | 42 |
|------------|----------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| Taxa | | | | | | | | | | | | | | | | | |
| 1 | insect 1 | - | - | - | C | T | C | C | T | G | C | A | C | C | G | T | C |
| 2 | insect 2 | - | - | - | - | T | C | G | T | G | T | A | C | C | A | T | G |
| 3 | insect 3 | T | G | G | C | T | C | C | T | G | C | A | C | A | G | T | G |
| 4 | insect 4 | T | G | G | T | T | C | C | T | G | C | A | C | G | G | T | A |
| 5 | insect 5 | T | G | G | T | T | C | C | T | G | C | A | C | T | G | T | C |
| 6 | insect 6 | T | G | G | C | T | C | C | T | G | C | A | C | C | G | T | A |
| 7 | insect 7 | T | G | G | A | T | C | C | T | G | C | A | C | T | G | T | G |

The **Plain Molecular** style, available for molecular sequence data, will color the cells by state, set the column width to be relatively narrow, and so on:

| Characters | | 27 | 28 | 29 | 30 | 31 | 32 | 33 | 34 | 35 | 36 | 37 | 38 | 39 | 40 | 41 | 42 |
|------------|----------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| Taxa | | | | | | | | | | | | | | | | | |
| 1 | insect 1 | - | - | - | C | T | C | C | T | G | C | A | C | C | G | T | C |
| 2 | insect 2 | - | - | - | - | T | C | G | T | G | T | A | C | C | A | T | G |
| 3 | insect 3 | T | G | G | C | T | C | C | T | G | C | A | C | A | G | T | G |
| 4 | insect 4 | T | G | G | T | T | C | C | T | G | C | A | C | G | G | T | A |
| 5 | insect 5 | T | G | G | T | T | C | C | T | G | C | A | C | T | G | T | C |
| 6 | insect 6 | T | G | G | C | T | C | C | T | G | C | A | C | C | G | T | A |
| 7 | insect 7 | T | G | G | A | T | C | C | T | G | C | A | C | T | G | T | G |

The **Nucleotide with AA Colors** style, available only for protein-coding nucleotide sequences, sets the cell color to represent the translated amino acid state, sets the columns to be narrow, hides the grid, and so on:

| Characters | | 2 2 8 5 6 3 3 3 4 5 6 7 8 9 0 1 2 |
|------------|----------|-----------------------------------|
| Taxa | | |
| 1 | insect 1 | - - - C T C C T G C A C C G T C |
| 2 | insect 2 | - - - - T C G T G T A C C A T G |
| 3 | insect 3 | T G G C T C C T G C A C A G T G |
| 4 | insect 4 | T G G T T C C T G C A C C G T A |
| 5 | insect 5 | T G G T T C C T G C A C T G T C |
| 6 | insect 6 | T G G C T C C T G C A C C G T A |
| 7 | insect 7 | T G G A T C C T G C A C T G T G |

The **Nucleotide with AA Translation** style, also available only for protein-coding nucleotide data, turns on the Show Translated AA's mode (see "[Coloring cells](#)" on page 259), and colors the translated AA:

| Characters | | 2 7 2 8 2 9 5 0 3 1 3 2 3 3 3 4 5 5 3 6 3 7 3 8 3 9 4 0 4 1 4 2 |
|------------|----------|---|
| Taxa | | |
| 1 | insect 1 | - - - C T C C T G C A C C G T C S C T V |
| 2 | insect 2 | - - - - T C G T G T A C C A T G - S C T M |
| 3 | insect 3 | T G G C T C C T G C A C A G T G G S C T V |
| 4 | insect 4 | T G G T T C C T G C A C G G G T A G S C T V |
| 5 | insect 5 | T G G T T C C T G C A C T G T C G S C T V |
| 6 | insect 6 | T G G C T C C T G C A C C G T A G S C T V |
| 7 | insect 7 | T G G A T C C T G C A C T G T G G S C T V |

The **Bird's Eye View** style, available for all data formats, hides the cell symbols, removes the grid, greatly reduces column width, and if the cells are uncolored, colors the cells by state:

| Characters | | | | | | | | | | | | | | | |
|------------|----------|--|--|--|--|--|--|--|--|--|--|--|--|--|--|
| Taxa | | | | | | | | | | | | | | | |
| 1 | insect 1 | | | | | | | | | | | | | | |
| 2 | insect 2 | | | | | | | | | | | | | | |
| 3 | insect 3 | | | | | | | | | | | | | | |
| 4 | insect 4 | | | | | | | | | | | | | | |
| 5 | insect 5 | | | | | | | | | | | | | | |
| 6 | insect 6 | | | | | | | | | | | | | | |
| 7 | insect 7 | | | | | | | | | | | | | | |

This one allows to see patterns in the matrix. If some other cell coloring is being used, for example by translated amino acid state, then that is retained:

| Characters | | | | | | | | | | | | | | | |
|------------|----------|--|--|--|--|--|--|--|--|--|--|--|--|--|--|
| Taxa | | | | | | | | | | | | | | | |
| 1 | insect 1 | | | | | | | | | | | | | | |
| 2 | insect 2 | | | | | | | | | | | | | | |
| 3 | insect 3 | | | | | | | | | | | | | | |
| 4 | insect 4 | | | | | | | | | | | | | | |
| 5 | insect 5 | | | | | | | | | | | | | | |
| 6 | insect 6 | | | | | | | | | | | | | | |
| 7 | insect 7 | | | | | | | | | | | | | | |

Finally, the **Wide with State Names** style, designed for nonmolecular data, expands the columns and turns on the **State Names in Cells** option:

| Characters | | 1 | 2 | 3 |
|------------|----------------|---------|------------|---------------|
| | | amnion | appendages | body covering |
| 1 | rayfinned fish | absent | fins | derm.scales |
| 2 | frogs | absent | legs only | smooth |
| 3 | turtles | present | legs only | epid.scales |
| 4 | lungfish | absent | fins | derm.scales |
| 5 | salamanders | absent | legs only | smooth |
| 6 | crocodiles | present | legs only | epid.scales |
| 7 | lizards | present | legs only | epid.scales |

Coloring characters

Shading character sets

Character sets (see ["Selecting or de-selecting sets of objects" on page 166](#) and ["Defining your own character sets" on page 201](#)) are sets of characters that are either predefined, like first codon positions in nucleotide data, or designated by the user. These can be highlighted in the editor by choosing items in the **Shade Char Sets** submenu in the **Display** menu. For instance, if you store a character set for skeletal characters and named it "skeletal", this will appear as an option in the **Shade Char Sets** submenu. Selecting it will gray each of the characters in the set:

| | | Characters | 1 | 2 | 3 | 4 | |
|---|-------------|------------|---------|--------------|-----------|--------------|---|
| | | Taxa | amnion | antorb fenes | lat.sphen | thermoreq. | |
| 2 | frogs | | absent | absent | not | poikilotherm | ▲ |
| 3 | turtles | | present | absent | not | poikilotherm | ☰ |
| 4 | lungfish | | absent | absent | not | poikilotherm | |
| 5 | salamanders | | absent | absent | not | poikilotherm | |
| 6 | crocodiles | | present | present | ossified | poikilotherm | |
| 7 | lizards | | present | absent | not | poikilotherm | |
| 8 | birds | | present | present | ossified | homeotherm | ▼ |

T 8 C 5

This shading should not be confused with selecting cells or blocks of cells in the matrix for purposes of editing or manipulation. This shading merely changes the background shade of cells for characters in the set. The effect of shading is added to other coloring effects (that is, you can have colored cells, shaded character sets, and colored codon positions in effect simultaneously).

By default, if you ask to shade characters in a character set, only the data cells themselves will be shaded. However, if you wish, you can alter the parts that are shaded, by checking or unchecking **Shade Character**

Numbers, Shade Character Names, and Shade Matrix Cells in the **Shade Char Sets** submenu in the **Display** menu. For example, if all of these are checked, the entire character column will be shaded:

| Characters | | 1 | 2 | 3 | 4 |
|------------|-------------|---------|--------------|-----------|--------------|
| Taxa | | amnion | antorb fenes | lat.sphen | thermoreq. |
| 2 | frogs | absent | absent | not | poikilotherm |
| 3 | turtles | present | absent | not | poikilotherm |
| 4 | lungfish | absent | absent | not | poikilotherm |
| 5 | salamanders | absent | absent | not | poikilotherm |
| 6 | crocodiles | present | present | ossified | poikilotherm |
| 7 | lizards | present | absent | not | poikilotherm |
| 8 | birds | present | present | ossified | homeotherm |

By default, the shaded character set elements are darkened, but by choosing **Lighten** from the **Shade Char Sets** submenu, they will instead be lightened:

| Characters | | 1 | 2 | 3 | 4 |
|------------|-------------|---------|--------------|-----------|--------------|
| Taxa | | amnion | antorb fenes | lat.sphen | thermoreq. |
| 2 | frogs | absent | absent | not | poikilotherm |
| 3 | turtles | present | absent | not | poikilotherm |
| 4 | lungfish | absent | absent | not | poikilotherm |
| 5 | salamanders | absent | absent | not | poikilotherm |
| 6 | crocodiles | present | present | ossified | poikilotherm |
| 7 | lizards | present | absent | not | poikilotherm |
| 8 | birds | present | present | ossified | homeotherm |

Shading character sets can sometimes allow you to more easily see patterns of character variation. For example, with a protein-coding nucleotide sequence, asking to lighten every third position makes the variation (or lack thereof) at first and second positions more obvious:

| | | Characters | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 30 |
|------|----------------|------------|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|----|
| Taxa | | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | |
| 1 | Homo sapiens | G | C | G | C | A | G | T | C | A | T | T | C | T | C | A | T | A | A | T | |
| 2 | Pan | G | C | G | C | A | T | T | A | T | C | C | T | C | A | T | A | A | T | | |
| 3 | Gorilla | G | C | G | C | A | G | T | T | G | T | T | C | T | T | A | T | A | A | T | |
| 4 | Pongo | G | C | G | C | A | A | C | C | A | C | C | C | T | C | A | T | G | A | T | |
| 5 | Hylobates | G | T | G | C | A | A | C | C | G | T | C | C | T | C | A | T | A | A | T | |
| 6 | Macaca fuscata | G | C | G | C | A | A | C | C | A | T | C | C | T | T | A | T | G | A | T | |

Coloring codon positions

If codon positions are specified for a protein-coding nucleotide sequence (see "[Specifying coding regions and codon positions](#) on page 292"), you can use the **Color Codon Positions** submenu of the **Display** menu to request that some indication of the positions be given in the editor, by coloring all or part of a character's column. For example, if you ask to **Color Char Numbers**, then the numbers of first position characters will be colored blue, second positions green, and third positions red:

| | | Characters | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 30 |
|------|----------------|------------|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|----|
| Taxa | | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | A | |
| 1 | Homo sapiens | G | C | G | C | A | G | T | C | A | T | T | C | T | C | A | T | A | A | T | |
| 2 | Pan | G | C | G | C | A | T | T | A | T | C | C | T | C | A | T | A | A | T | | |
| 3 | Gorilla | G | C | G | C | A | G | T | T | G | T | T | C | T | T | A | T | A | A | T | |
| 4 | Pongo | G | C | G | C | A | A | C | C | A | C | C | C | T | C | A | T | G | A | T | |
| 5 | Hylobates | G | T | G | C | A | A | C | C | G | T | C | C | T | C | A | T | A | A | T | |
| 6 | Macaca fuscata | G | C | G | C | A | A | C | C | A | T | C | C | T | T | A | T | G | A | T | |

The **Color Char Names** option will color the character name cells, and the **Color Matrix Cells** option will color the cells containing data. In the following matrix, both character numbers and matrix cells are colored:

| Characters | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|------------|----------|---|---|---|---|---|---|---|---|---|----|----|----|
| Taxa | | - | - | - | - | - | - | - | - | - | - | - | - |
| 1 | insect 1 | - | - | - | - | - | - | - | - | - | - | - | - |
| 2 | insect 2 | - | - | - | - | - | - | - | - | - | - | - | - |
| 3 | insect 3 | - | - | - | G | A | G | T | G | C | A | A | G |
| 4 | insect 4 | - | - | - | G | A | T | G | T | A | A | A | G |
| 5 | insect 5 | - | - | - | - | - | - | - | - | - | A | A | G |
| 6 | insect 6 | - | - | - | - | - | - | - | - | - | - | - | - |
| 7 | insect 7 | - | - | - | G | A | T | T | G | T | A | A | G |

The effect of coloring codon positions is added to other coloring effects (that is, you can have colored cells, shaded character sets, and colored codon positions in effect simultaneously).

Other display options

Changing the width of columns

Change the width of the leftmost column (that listing taxon names, or, for a transposed matrix, that listing character names) by positioning the cursor over the black vertical line at the right edge of the column. The

cursor will change to a :

| Characters | | 1 | 2 | 3 |
|------------|----------------|------|-------|------|
| Taxa | | amni | upper | body |
| 1 | rayfinned fish | 0 | 0 | 0 |
| 2 | frogs | 0 | 1 | 1 |
| 3 | turtles | 1 | 1 | 2 |
| 4 | lungfish | 0 | 0 | 0 |

Changing the width of the first column

Click, and drag the line to its new position.

There are two methods to change the width of other columns. The **Column Width** submenu in the **Display** menu provides some standard column width settings you might use. There is also a shortcut. Hold down the Control key, and move the cursor between two column numbers; the cursor will change to the column

width adjust tool (). This tool will adjust the width of the column immediately to the left of the tool (and, as all columns in MacClade must be the same width, it will adjust all the other columns to be that width as well). For example, if you grabbed between column 1 and 2,

| 1 | 2 | 3 | 4 | 5 |
|------|-------|------|-------|--------|
| amni | upper | body | therm | int.no |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 | 1 |
| 1 | 1 | 2 | 0 | 1 |
| 0 | 0 | 0 | 0 | 1 |

and dragged the column boundary to the right an additional column width,

| 1 | 2 | 3 | 4 | 5 |
|------|-------|------|-------|--------|
| amni | upper | body | therm | int.no |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 | 1 |
| 1 | 1 | 2 | 0 | 1 |
| 0 | 0 | 0 | 0 | 1 |

then the columns would be expanded to be about twice as wide:

| 1 | 2 | |
|--------|------------|------|
| amnion | appendages | body |
| 0 | 0 | |
| 0 | 1 | |
| 1 | 1 | |
| 0 | 0 | |

When you change column width, MacClade will readjust the window size so that only whole columns are shown.

MacClade may change column width automatically if you select one of the **Data Matrix Styles** in the **Display** menu.

Showing or hiding the grid and column and row numbers

Under the **Show** submenu of the **Display** menu you can select items to show or hide various elements in the data editor. This includes the grid lines shown in the matrix, and the taxon and character numbers.

Fonts

Change fonts and type sizes of the whole data editor using the **Font** and **Size** submenus in the **Display** menu.

The names of taxa can also be individually italicized. Select the name or names to be italicized, and select **Italic** from the **Display** menu. If the names are already italic, they will revert to nonitalic.

Choosing character by taxa or taxa by characters

To transpose the matrix (Characters X Taxa as opposed to Taxa X Characters), choose **Transposed Matrix**

under the **Display** menu. A matrix transposed on the screen will also be written into a NEXUS file as a transposed matrix. An example of an untransposed and a transposed matrix is shown below.

| | | Characters | 1 | 2 | 3 | 4 | |
|---|---------------|------------|-------|-------|-------|-------|---|
| | | Taxa | silve | inter | inter | inter | |
| 1 | balli | | 1 | 1 | 1 | 0 | ▲ |
| 2 | foveum | | 1 | 1 | 1 | 0 | ☰ |
| 3 | argenteolum | | 1 | 3 | 2&3 | 0&1 | |
| 4 | alaskense | | 1 | 3 | 2&3 | 0 | |
| 5 | semenovi | | 1 | 2 | 2 | 0 | |
| 6 | stenoderum | | 1 | 3 | 3 | 0&2 | |
| 7 | carinula | | 1 | 3 | &1&2 | 0 | ▼ |

T 10 C 5

A matrix in normal (untransposed) orientation,
with each character occupying a column

| | | Taxa | 1 | 2 | 3 | 4 | |
|---|------------------------|------------|-------|--------|-------------|-----------|---|
| | | Characters | balli | foveum | argenteolum | alaskense | |
| 1 | silver spots | | 1 | 1 | 1 | 1 | ▲ |
| 2 | interval 3 micro. | | 1 | 1 | 3 | 3 | ☰ |
| 3 | interval 4+5 micro. | | 1 | 1 | 2&3 | 2&3 | |
| 4 | interval 6+7 micro. | | 0 | 0 | 0&1 | 0 | |
| 5 | outer mirror | | - | - | - | - | |
| 6 | silver spot | | 0 | 0 | 0 | 0&1 | |
| 7 | elytral microsculpture | | 0 | 0 | 0 | 0 | ▼ |

T 3 C 8

A transposed matrix, with each character occupying a row

Footstates

Turning on the footstates feature in the **Display** menu will display a list of state names (if any have been defined using the **State Names & Symbols** window, and for protein data) of the character of the currently selected cell in the box at the bottom.

| | | Characters | | | | | |
|------|----------------|------------|-------|------|-------|-------|-------|
| | | 1 | 2 | 3 | 4 | 5 | 6 |
| Taxa | | amni | upper | body | therm | int.n | atria |
| 1 | rayfinned fish | 0 | 0 | 0 | 0 | 0 | 0 |
| 2 | frogs | 0 | 1 | 1 | 0 | 1 | 1 |
| 3 | turtles | 1 | 1 | 2 | 0 | 1 | 1 |
| 4 | lungfish | 0 | 0 | 0 | 0 | 1 | 1 |
| 5 | salamanders | 0 | 1 | 1 | 0 | 1 | 1 |
| 6 | crocodiles | 1 | 1 | 2 | 0 | 1 | 1 |
| 7 | lizards | 1 | 1 | 2 | 0 | 1 | 1 |
| 8 | birds | 1 | 2 | 3 | 1 | 1 | 1 |
| 9 | mammals | 1 | 1/2 | 4 | 1 | 1 | 1 |

0: poikilotherm 1: homeotherm

Data editor displaying footstates

If you have footstates turned on, you do not have access to footnotes.

If the footstates information is too long to fit in the box at the bottom, you can expand the box by grabbing its top border and dragging it up.

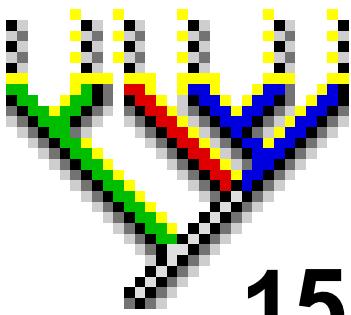
NOTE: The Pop-up data entry tool (page 234) also provides a list of named states in a character.

Adjusting the color of the selected block of cells

You can change the color used by MacClade to indicate a selected block of cells. By default, the color is gray. Using the **Set Editor Selection Color** in the **Display** menu, you can change the color to dark gray, red, orange, green, and so on. For example, here are three of the available colors:

| | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 1 |
| 1 | 1 | 2 | 0 | 1 | 1 | 1 | 2 | 0 | 1 | 1 | 1 | 2 | 0 | 1 |
| 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 |
| 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 1 |
| 1 | 1 | 2 | 0 | 1 | 1 | 1 | 2 | 0 | 1 | 1 | 1 | 2 | 0 | 1 |

You may find this useful if you have difficulty seeing the default gray color.



ASSUMPTIONS ABOUT CHARACTERS

15

This chapter describes assumptions you can apply, and how to apply them, in MacClade. Some of the theoretical background for this chapter is found in [Chapter 3](#) and [Chapter 4](#), which discuss transformation assumptions in detail.

Displaying and changing current assumptions

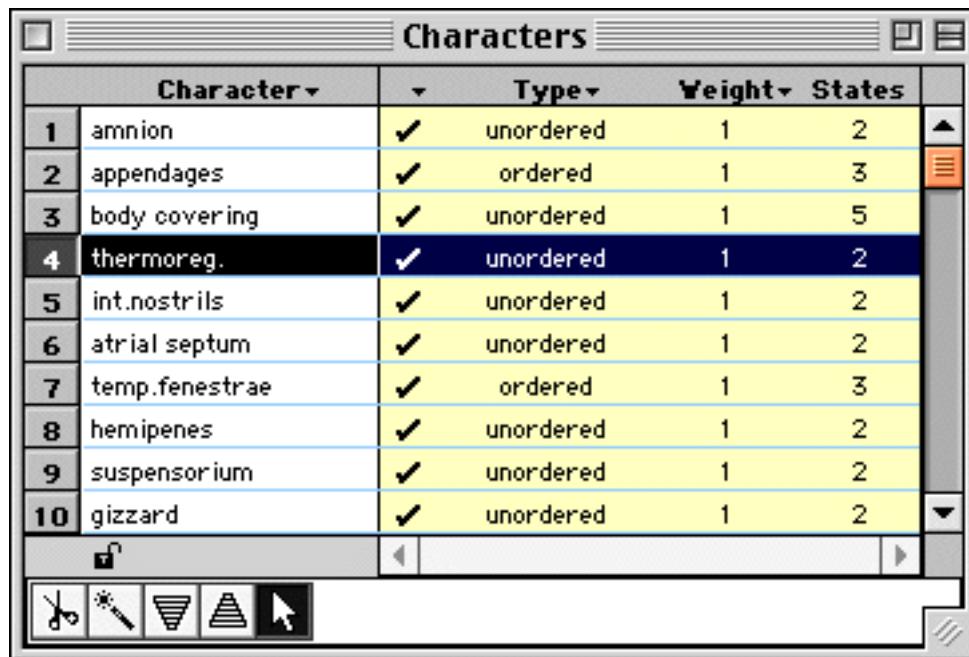
To see the assumptions currently applied to characters in your data matrix, look at columns in the character list window. To change attributes of a character, you first need to select that character. You can do this in one of three ways:

1. Select the row corresponding to the character in the character list window,
2. Select the entire column or row corresponding to the character in the data editor, or
3. Trace the character on a tree, and have the tree window frontmost.

Once characters are selected, you can use the various submenus in the **Characters** menu to change the attributes of these characters. These procedures are detailed in subsequent sections.

Character list window

The character list window lists the characters in the data matrix, along with various assumptions and statistics that apply to each of them:



| Character | Type | Weight | States |
|------------------|-------------|--------|--------|
| 1 amnion | ✓ unordered | 1 | 2 |
| 2 appendages | ✓ ordered | 1 | 3 |
| 3 body covering | ✓ unordered | 1 | 5 |
| 4 thermoreg. | ✓ unordered | 1 | 2 |
| 5 int.nostriils | ✓ unordered | 1 | 2 |
| 6 atrial septum | ✓ unordered | 1 | 2 |
| 7 temp.fenestrae | ✓ ordered | 1 | 3 |
| 8 hemipenes | ✓ unordered | 1 | 2 |
| 9 suspensorium | ✓ unordered | 1 | 2 |
| 10 gizzard | ✓ unordered | 1 | 2 |

It is displayed by selecting **Character List** in the **Characters** menu. This window serves two purposes: to inform the user of current assumptions and results, and to allow the user to change current assumptions. Thus, characters can be included and excluded, and their weights and types changed, in this table.

Information on how to select characters in the character list window, so that their properties might be changed, is given in [Chapter 10](#).

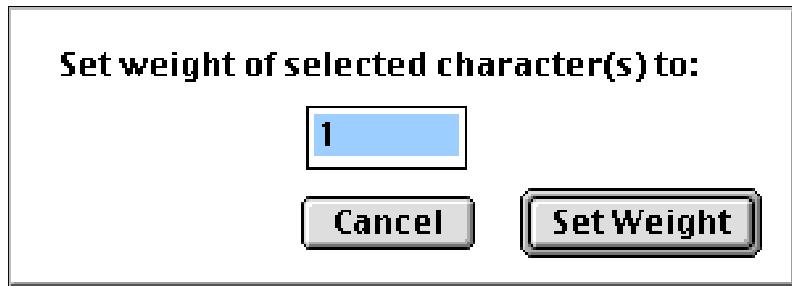
Character weights

Changing character weights

MacClade allows you to weight characters differentially, so that more heavily weighted characters contribute more to the treelength. (See [page 374](#) concerning weighting and treelength; for discussion of weighting see Farris, 1969, 1983; Felsenstein, 1981a; Neff, 1986; Q. Wheeler, 1986; W. Wheeler, 1990.) There are several ways you can change the weight of a character.

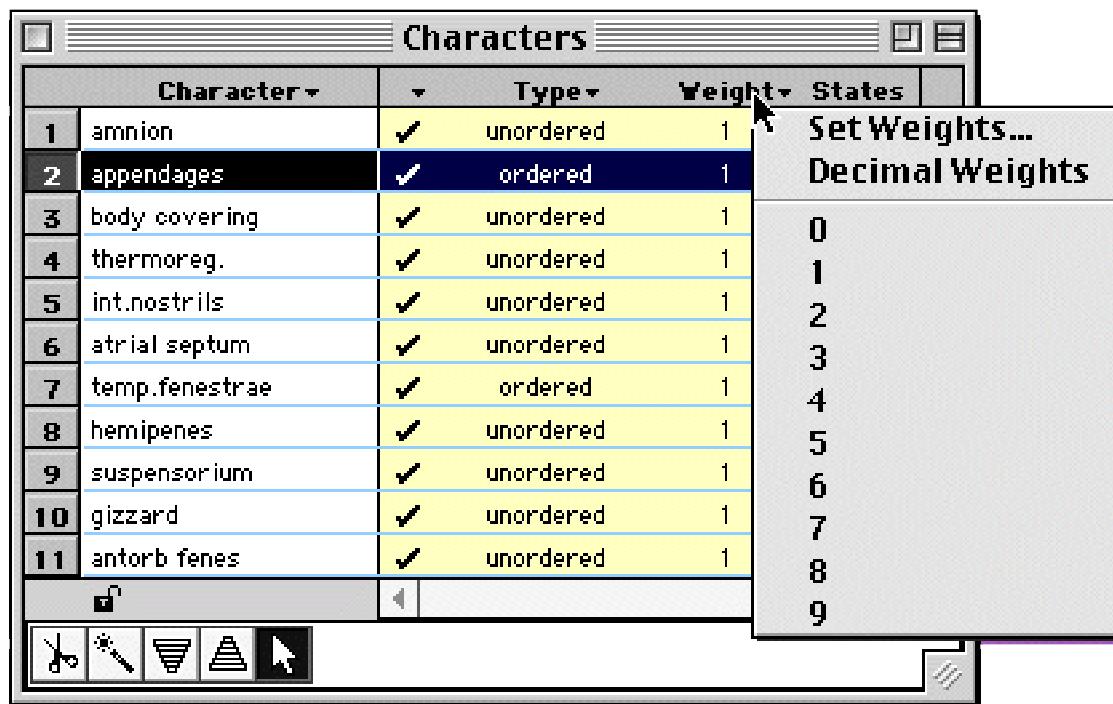
Weights of one or more characters can be changed by selecting the characters, either in the character list window, or in the data editor, or by tracing the character, and then selecting a number from 0 to 9 in the **Change Weight** submenu in the **Characters** menu. This will change the weight of the selected characters to the number chosen.

The **Set Weight** dialog box in the **Change Weight** submenu of the **Characters** menu allows you more control of the change in weights. In the dialog box, enter the new weight. Weights can be integral (0 through 9999) or decimal (0.00 through 99.99).



You cannot mix integral weights with decimal weights; all characters must be weighted in one fashion. To convert integral weights to decimal weights, select **Decimal Weights** in the **Change Weights** submenu of the **Characters** menu. To convert decimal weights to integral weights, reselect the same menu item. In converting decimal to integral weights, MacClade gives you two options: simply to truncate (e.g., 3.17 becomes 3) or to multiply by 100 first (3.17 becomes 317). The purpose of the latter option is to preserve accuracy if you must convert weights to integral. Similarly, in converting integral to decimal weights, MacClade gives you two options, to simply append a decimal (e.g., 61 becomes 61.0) or to divide by 100 first (61 becomes 0.61).

If you are changing weights in the character list window, then the **Change Weights** submenu is also available if you click on the column title "Weights" in the window, as shown below:

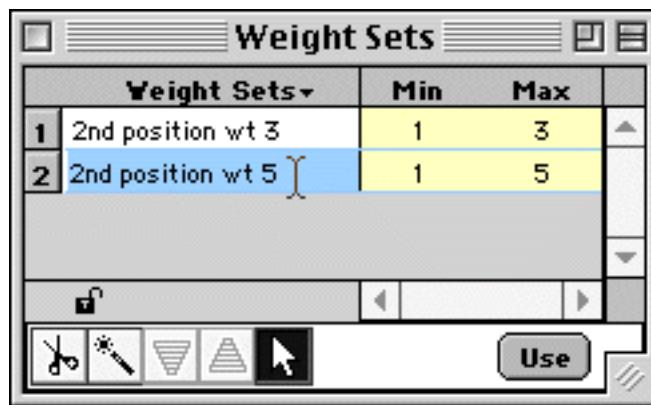


NOTE: Double-clicking on the title of the Weight column in the table will display the Set Weights dialog box.

Weight sets

A weight set is a list of assigned weights to various characters. Weight sets can be stored to save a particular weighting scheme for all characters, then recalled later when necessary.

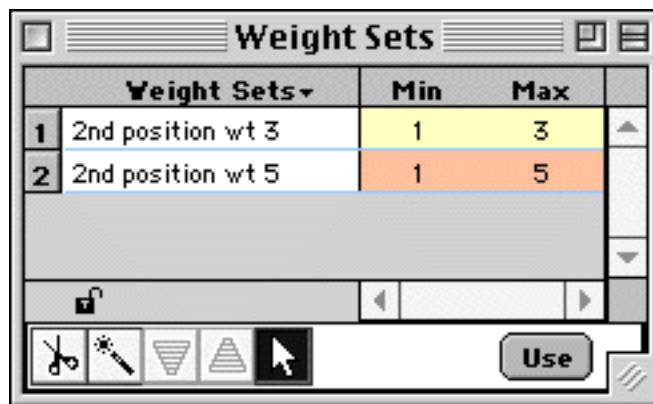
There is one predefined weight set (**All Equal Weights**, available under the **Weight Sets** submenu of the **Characters** menu), and you can define your own. The current weight settings for the characters can be named and stored by choosing **Store Weight Set** from the **Weight Sets** submenu of the **Characters** menu. This will bring forth the Weight Sets list window, in which you can name the new weight set:



To invoke already-stored weight sets, choose **Weights Sets List** from the **Weight Sets** submenu of the **Characters** menu. In the Weight Sets list window that appears, select the weight you wish to invoke, and press the **Use** button:



The current weight set will be indicated by orange shading:

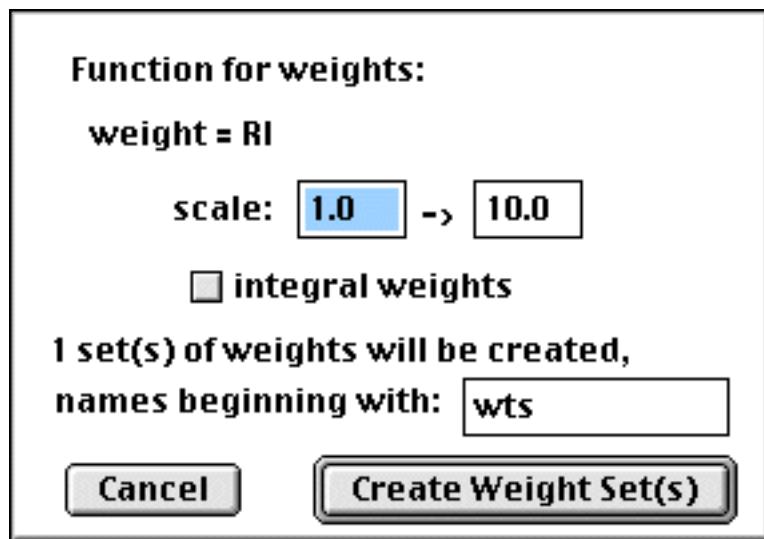


Creation of weight sets based on calculated indices for each character

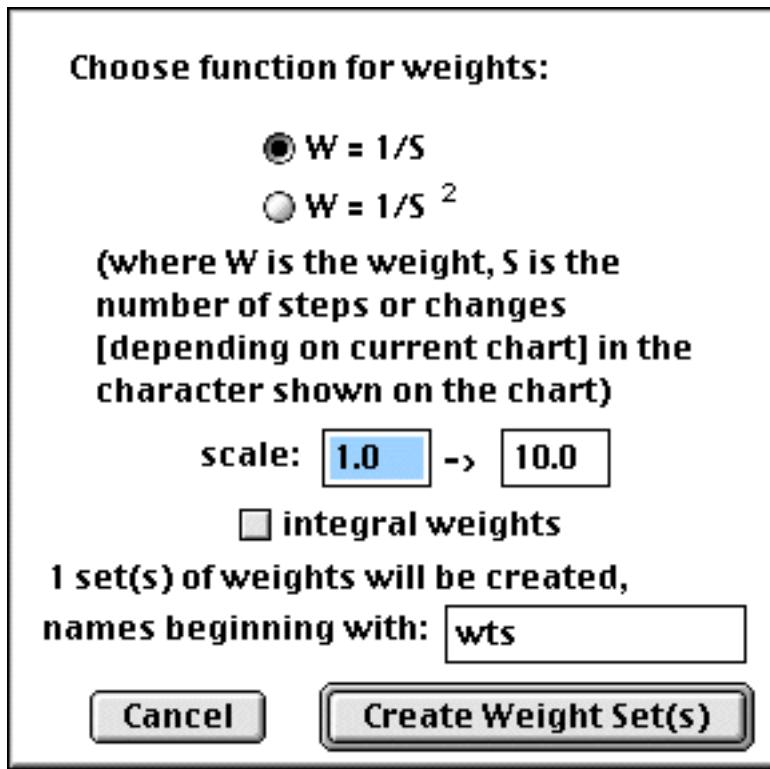
If you create a chart of steps, changes, CI, RI, or RC for each character (["Number of steps/etc. on each character in sequence" on page 392](#)), or for codon positions for nucleotide data (["Number of steps/etc. by codon position" on page 392](#)), you can ask MacClade to calculate weights for each character based on the value shown for each character, using the **Chart to Weight Set** menu item in the **Chart** menu. For example, if you had asked MacClade to plot RC on the vertical axis and character number on the horizontal, and then asked MacClade to convert this to a weight set, MacClade would convert the RC for each character into a weight value. If codon position is on the horizontal axis, then MacClade will calculate a weight for each character according to its codon position, with all first positions receiving the same weight, and so on.

Note that MacClade does **not** invoke the created weight set or sets unless you so request using the weight sets list window.

The dialog box presented to you when you choose **Chart to Weight Set** allows you to choose exactly how the conversion takes place; the options you will be given will depend upon the index charted. If the statistic displayed in the chart is CI, RI, or RC, then you will be presented with the following dialog box:



Otherwise, the following will appear:



These differ primarily in the functions available to convert the value of the statistic to a weight. If the statistic displayed is CI, RI, or RC, then MacClade will convert the index values directly to weights. If the statistic being displayed is steps or changes, then the weights created will be a function of that statistic. Two functions are available: the inverse of the number of steps or changes, or the squared inverse (Williams and Fitch, 1989, 1990).

You can ask MacClade to scale weights so that they fit within the range you request. For example, if you want weights scaled so that a CI of 0 corresponds to a weight of 1.0, and a CI of 1 corresponds to a weight of 10.0, then enter 1.0 and 10.0 into the two scale boxes. If you enter a value that is above the maximum allowable by MacClade, the program will beep.

Characters in any interval with 0 observed steps or changes in a chart showing steps or changes will be assigned a weight equal to the maximum value of the scale chosen. Thus, 0 steps using the $W = 1/S$ weight function and a weight scale from 1 to 10 will be assigned a weight of 10, not a weight of ∞ . Characters with the next-lowest number of steps or changes will be assigned the same, maximal value.

If you select integral weights, then MacClade will round weights to the nearest integer.

You can give a name to the weight set created by entering a name in the appropriate field in the dialog box. In the dialog box illustrated, the name is "wts". If for each character there is more than one value of the statistic (perhaps as there is more than one tree) then MacClade will calculate a weight set based on the minimum values (which, in this example, would be called "wts min"), another based on the average values ("wts avg"), and so on.

These weight sets can be used in a Successive Approximations Character Weighting (SACW) analysis (Farris, 1969, 1988; Carpenter, 1988). SACW involves a search for the most parsimonious tree(s), production of a weight set based on the CI, RI, RC, or whatever of each character on the tree(s), re-searching for the most parsimonious tree(s) with each character weighted according to the weight set, and so on, until the trees found in two successive iterations are identical. An adequate justification for this procedure has yet to be provided (D. Maddison, 1994). Some aspects of the method are not well defined, such as which of the multiple weight sets created by MacClade (due to multiple parsimonious trees, for example) you should use. We do not necessarily advocate use of SACW, but we have placed this capability in MacClade to allow you to explore the behavior of this method.

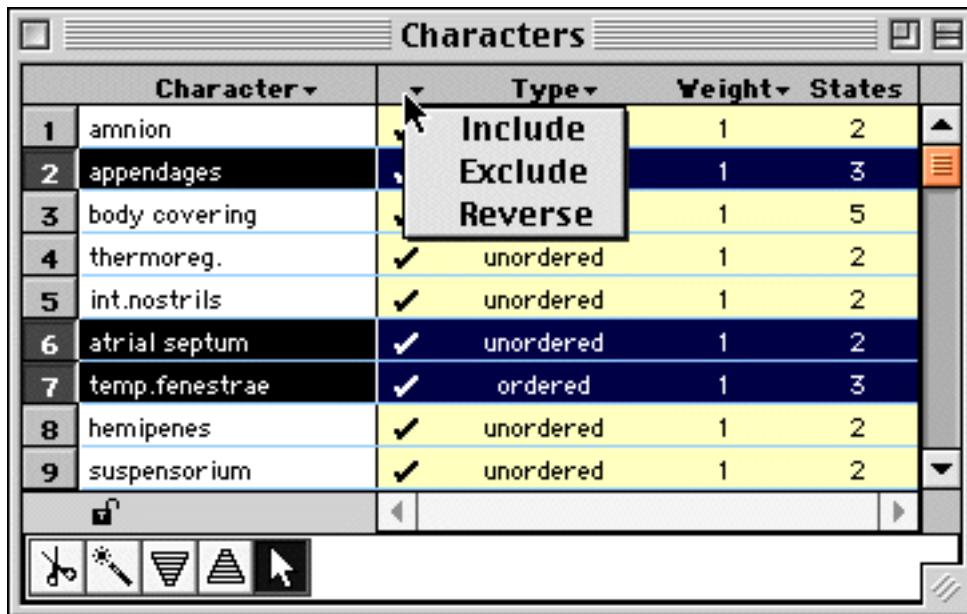
Character exclusion

A more drastic form of weighting is exclusion of characters.

Including and excluding characters

Characters in the data matrix can be included or excluded from any analysis. To include or exclude a character, select it in the character list window or in the data editor, and choose **Include**, **Exclude**, or **Reverse** from the **Include-Exclude Chars** submenu of the **Characters** menu. **Include** and **Exclude** do the obvious; **Reverse** includes previously excluded characters, and excludes previously included characters. Included characters are indicated by at the left side of their rows in the character list window; excluded characters are indicated by .

If you are using the character list window to exclude or include characters, then the **Include-Exclude Chars** submenu is also available if you touch on the little triangle in the title region of the column indicating character inclusion or exclusion, as shown below:



Behavior of excluded characters

Excluded characters, for the purposes of MacClade's calculations, don't exist. That is, they are not included in any calculations and you can't trace their evolution. This means that you can have user-defined type characters with polytomous trees, normally not allowed in MacClade, as long as all such characters are excluded. Of course, the excluded characters are not deleted from the data files (unless you explicitly ask MacClade to delete them), and can be included once again when appropriate.

Inclusion sets

An inclusion set is a list of those characters that are to be included by MacClade in any calculations. Sets of included characters can be saved and recalled in a fashion similar to weight sets (see ["Weight sets" on page 275](#)), using items in the **Inclusion Sets** submenu of the **Characters** menu and the **Inclusion Sets** list window.

There are three predefined inclusion sets: **All Included**, **All Excluded**, and **Exclude Uninformative**. Choosing **All Included** will cause all characters to be included; choosing **All Excluded** will cause all to be excluded, and choosing **Exclude Uninformative** will cause all characters uninformative for parsimony analyses on the current tree to be excluded, as described in the next section.

NOTE: MacClade calls the sets indicating inclusion and exclusion "inclusion sets", so that the user can name them positively (e.g., "larval characters") to indicate what is included. However, they are actually saved in the NEXUS file as lists of excluded characters.

Exclude Uninformative

If **Exclude Uninformative** is selected from the **Inclusion Sets** submenu of the **Characters** menu when the tree window is visible, MacClade scans the characters to see which ones are uninformative. A character is uninformative if, according to its current transformation type, any possible dichotomous tree would

require the same number of steps in the character. *Only unordered and ordered characters are examined.* Also, only currently included characters are examined; excluded characters will remain excluded.

NOTE: A character is considered uninformative only if it is irrelevant for choosing among trees using parsimony. The character may still be informative for other inferences (e.g., for inferring the relative frequencies of change between states using the **State Changes & Stasis** chart). "Uninformative" characters may also be informative for choosing between trees using maximum likelihood estimation or other techniques not based on parsimony.

Remember that whether a character is uninformative depends on the assumptions you use. Therefore, if you exclude an unordered character as uninformative, you should recheck whether it is uninformative if you later change it to ordered.

MacClade determines whether a character is uninformative by calculating the minimum and maximum conceivable lengths for the character. (See discussion regarding consistency and retention indices in [Chapter 4](#) and [Chapter 19](#).) If the minimum and maximum are the same, then the character is uninformative. Because MacClade can calculate both the minimum and maximum only for unordered and ordered characters, **Exclude Uninformative** functions for characters of only these types.

MacClade's **Exclude Uninformative** is calculated over the taxa in the currently displayed tree, and thus is available only from the tree window! A character that is uninformative on a tree with a subset of the taxa in the data file may become informative when all taxa are included in the tree.

If you want to delete uninformative characters completely from the data file, first make sure all characters are included. Then, exclude the uninformative ones using **Exclude Uninformative** in the tree window, open the character list window, select the excluded characters (using either the Selection Wand or the **Select** submenu; see ["Selecting objects" on page 163](#)), and press the Delete key to delete the selected characters. Before you do this, however, remember that a character's uninformativeness depends on the taxa in the tree and the transformation type of the character. For molecular sequence data, you may want to use **Fill** to name characters by their sequence positions before deletion (see ["Naming characters" on page 193](#)), so that the original sequence positions can be known after deletion.

Excluding characters with a certain percentage of missing data or gaps

The **Show % Missing** or **Show % Gaps** feature (["Calculating the percentage of missing data and gaps" on page 379](#)) can be used in conjunction with the Selection Wand (["Selecting similar objects" on page 170](#)) to select all characters with a certain fraction of missing data or gaps, which could then be altered as desired. For example, imagine you wished to exclude all sites at which more than 25% of the sequences had gaps. You could turn on **Show % Gaps**, and the character list window would appear with a new column:

| Site | Pos | Type | Weight | States | % Gaps |
|------|-----|-----------|--------|--------|--------|
| 20 | - | unordered | 1 | 1 | 82.8 |
| 21 | - | unordered | 1 | 1 | 80.8 |
| 22 | - | unordered | 1 | 3 | 78.8 |
| 23 | - | unordered | 1 | 3 | 44.4 |
| 24 | - | unordered | 1 | 1 | 25.3 |
| 25 | - | unordered | 1 | 1 | 19.2 |
| 26 | - | unordered | 1 | 1 | 17.2 |
| 27 | - | unordered | 1 | 2 | 16.2 |

If you use the Selection Wand, and touch on the "25.3" in the % Gaps column of site 24, MacClade will select all sites with 25.3% gaps; but if you hold down the Option key when you do this, MacClade will select all sites with at least 25.3% gaps. These can then be excluded:

| Site | Pos | Type | Weight | States | % Gaps |
|------|-----|------------|--------|--------|--------|
| 20 | - | (excluded) | 1 | 1 | 82.8 |
| 21 | - | (excluded) | 1 | 1 | 80.8 |
| 22 | - | (excluded) | 1 | 3 | 78.8 |
| 23 | - | (excluded) | 1 | 3 | 44.4 |
| 24 | - | (excluded) | 1 | 1 | 25.3 |
| 25 | - | unordered | 1 | 1 | 19.2 |
| 26 | - | unordered | 1 | 1 | 17.2 |
| 27 | - | unordered | 1 | 2 | 16.2 |

Protein-coding regions

For DNA and RNA data, you can specify which bases are in protein-coding regions, and which bases are the first, second, or third bases of a codon. See ["Specifying coding regions and codon positions" on page 292](#) for further details.

Character transformation assumptions

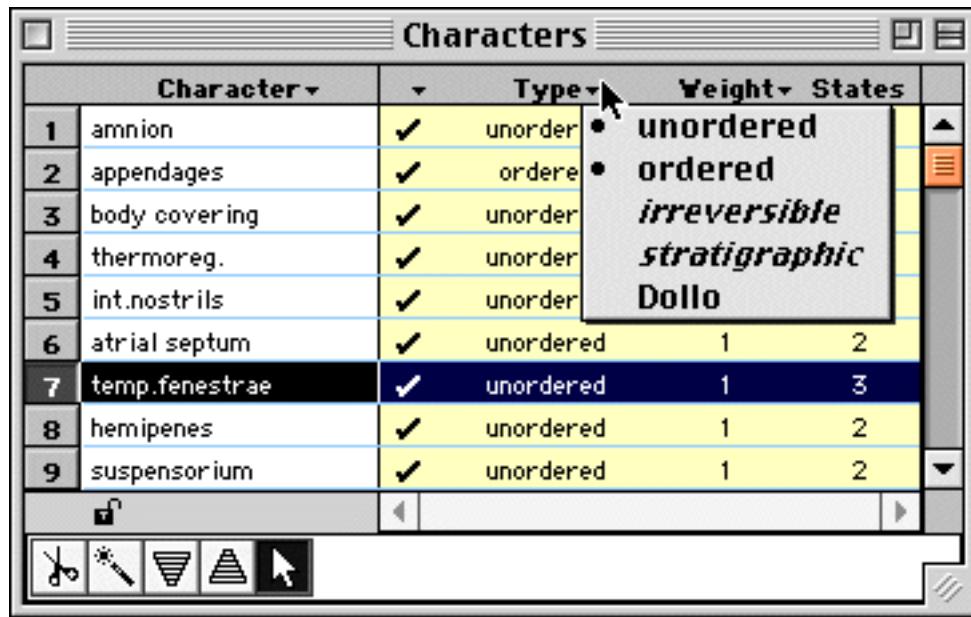
Transformation types

In MacClade, assumptions about how characters evolve are applied by assigning a particular transformation type to a character. Transformation types specify how many steps it costs to go from one state to any other state, and if some transformations are impossible. Transformation types are described in more detail in [Chapter 4](#).

MacClade has both predefined and user-defined transformation types. The predefined types are unordered, ordered, irreversible, stratigraphic, and Dollo. For user-defined transformation types, the user must first define the transformation type, and then apply it to the character.

Applying transformation types to characters

The type of a character can be specified by selecting the character, either in the character list window (see ["Selecting objects" on page 163](#)), or in the data editor, or by tracing the character, and then selecting the name of the desired type in the **Change Type** submenu in the **Characters** menu. The **Change Type** menu also appears if you touch the "Type" heading in the character list window:



This submenu gives as options all available types (at minimum, unordered, ordered, irreversible, stratigraphic, and Dollo). Type names marked with a black circle are in current use. If some of the type names are dimmed (unavailable), then those types are currently prohibited by the situation. For example, the Dollo type cannot be selected if the tree in the tree window has polytomies. For information about selecting elements in the window, see ["Selecting objects" on page 163](#).

Any transformation types that are inherently directed or polarized (that is, in which there exists at least one pair of states x and y such that the number of steps going from x to y is not the same as from y to x) are *italicized* in the **Change Type** submenu. These characters may provide some preference for possible roots of the tree. If such a type is currently in use, then a small tree trunk with roots appears beside the message box at the lower left of the tree window.

Types can also be assigned to characters using type sets, as described below.

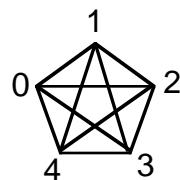
The default transformation type for MacClade is unordered. That is, if you fail to specify a transformation type for a character, MacClade will assume it is of unordered type. MacClade uses this type as default because it is considered to embody weaker, more conservative assumptions than other types. For instance, designating a character as ordered will often increase the number of steps; it can never decrease it. Unless there is good reason to support the assumption that characters are ordered, it seems better to make the more conservative assumption and to use unordered characters. Of course, if you have good evidence for a transformation type other than unordered, you should use it. See [Chapter 3](#) for more discussion of the use of assumptions.

A user can change a data file's default transformation type by selecting the name of the type in the **Type Edit** dialog box and clicking on the button **Set Default**. After this, any new characters created are initially assigned this default type. Remember, this is the default type for the current file, not the preference for the default type. To change the preference for the default type, set the file's default type, and then use the **Save Preferences** dialog box of the **File** menu.

Predefined transformation types

MacClade has five predefined transformation types: **unordered**, **ordered**, **irreversible**, **stratigraphic**, and **Dollo**. These predefined transformation types can be briefly characterized as follows; they are discussed in more detail in [Chapter 4](#). The section "[The ordering of character states](#)" on page 284 should be consulted for further explanation regarding ordered, irreversible, stratigraphic, and Dollo characters.

1. **Unordered** ("Fitch parsimony"; Fitch, 1971; Hartigan, 1973) — A change from any state to any other state is counted as one step.



Unordered

2. **Ordered** ("Wagner parsimony"; Farris, 1970, Swofford and Maddison, 1987) — The number of steps from one state to another state is counted as the (absolute value of the) difference between their state numbers.

0 — 1 — 2 — 3 — 4

Ordered

3. **Irreversible** ("Camin-Sokal parsimony"; Camin and Sokal, 1965) — The number of steps from one state to another state is counted as the difference between their state numbers, with the restriction that decreases in state number do not occur.

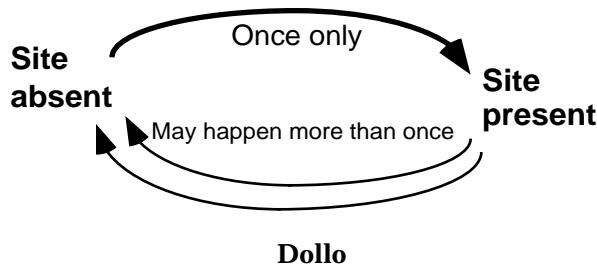
0 → 1 → 2 → 3 → 4

Irreversible

4. **Stratigraphic** (Fisher's, 1982, 1988, 1992 "stratigraphic parsimony") — Characters assigned this type are treated as indicating stratigraphic age, with each state representing a stratum, or interval of time. Older through younger strata are coded by states of smaller through larger state numbers,

respectively. The reconstruction of states at ancestral nodes that MacClade provides is in some sense a reconstruction of the ages of ancestors, though it should be noted that when MacClade shades a hypothetical ancestor as having state i in a stratigraphic character, this does not mean that this ancestor occurred in stratum i , but rather that it occurred *between* strata $i - 1$ and i . As a default, one step is counted for each absence in a stratum, but as noted below in the section "Defining and editing types", the user can vary the cost of absences in particular strata. Stratigraphic characters behave very much as do irreversible characters, differing mostly in that the cost of each "step" can be varied and in their behavior when observed taxa are fixed as ancestors. This character type is discussed in more detail in [Chapter 4](#).

5. **Dollo** (Farris, 1977) — The number of steps from one state to another state is counted as the difference between the numbers given to the states, with the restriction that each increase in state number occurs only once.



Dollo characters can include only three states for standard format data files. These states are by default 0–2. For extended standard and protein data files, states 0–7 are allowed. Dollo characters are not allowed with DNA and RNA format files. MacClade 4 assumes, as did older versions of MacClade (Maddison and Maddison, 1987, 1992), what Swofford (1991, 2000) calls the "unrooted" Dollo assumption. That is, MacClade does not assume that there was *exactly one* gain from state 0 to state 1; it assumes only that there was *no more than one* gain from 0 to 1, no more than one from 1 to 2, and so on.

The ordering of character states

For ordered, irreversible, stratigraphic, and Dollo characters there is an implicit ordering or sequence of states. This ordering is used to calculate how many steps are required to go between one state and another; thus for an ordered character it is one step to change 0 to 1, and one to change 1 to 2, but two steps to change 0 to 2, because they are ordered in the sequence 0-1-2. In the above discussion we talked of states 0, 1, 2 and so on, but what is the ordering for a DNA data file with states A, C, G, T, or a data file for which you have defined the symbols to be L, C, J, X, P? The order MacClade assumes depends on the sequence of symbols (see the section "[Symbols for states](#) on page 195"). If the **State Names & Symbols** window lists A, B, C, D as the symbols in that order for states numbered 0 through 3, then MacClade assumes the order of states is A-B-C-D. But if in the **State Names & Symbols** window you define the symbols to be A, C, D, and B, in that order, MacClade will assume that to be the ordering for use with ordered, irreversible, Dollo, and stratigraphic characters.

See the section "[State numbers vs. symbols vs. names](#)" on page 198 for a discussion of how you can change the ordering of character states.

User-defined transformation types

In addition to the predefined transformation types discussed above, MacClade users can specify their own assumptions. This can be done by creating and editing new types in the **Type Edit** dialog box, then applying these new types to particular characters.

MacClade's user-defined types can take one of two forms: step matrices or character state trees. Step matr-

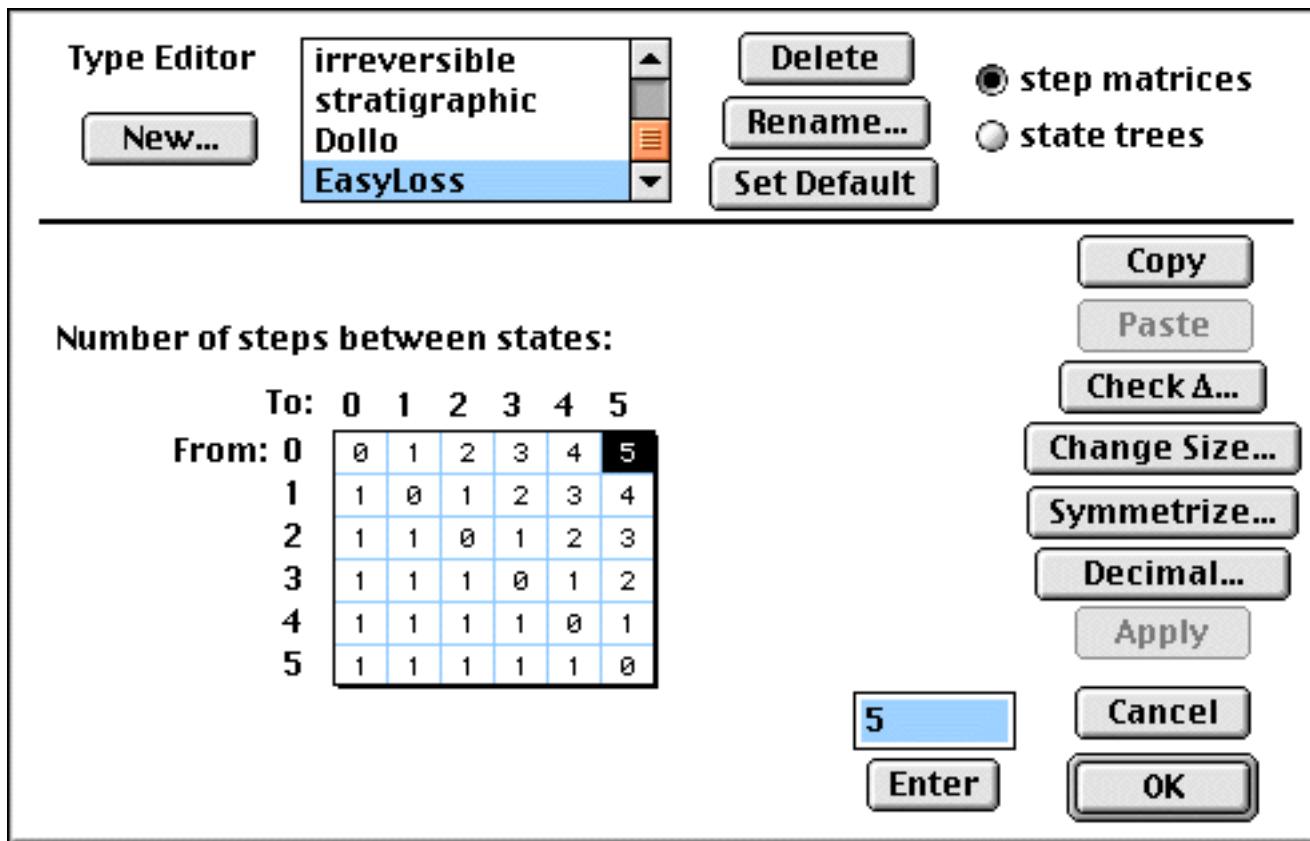
ces specify, for each pair of states, the number of steps for transformations between these states. A step matrix might specify, for example, that a 0 to 1 transformation costs 1 step, but a 1 to 0 transformation costs 3 steps. Character state trees indicate graphically how transformations between states are constrained to a treelike graph. Both step matrices and character state trees are discussed in more detail in [Chapter 4](#) and [Chapter 3](#).

Step matrices allow you to use a great variety of assumptions. Indeed, it is even possible to define step matrices that indicate the same assumptions as the predefined types unordered, ordered and irreversible, as shown on [page 72](#). However, if you want to use these same assumptions, it is much better to use the predefined types than to build user-defined equivalents, for MacClade handles the built-in types much more quickly and more flexibly than user-defined types. User-defined types have a number of limitations, as discussed below.

The means by which you can create and edit matrix types and character state tree types are described in the next section.

Defining and editing types

User-defined transformation types are defined independent of any particular characters, and are later applied to the characters of your choice. They can be defined and edited in the **Type Edit** dialog box (available in the **Characters** menu). This dialog lists the currently defined types, and allows you to edit existing user-defined types and create new ones. The dialog box looks something like this:



In the list of types, those marked by a bullet (•) are in active use (that is, at least one included character is currently of that type). The one with an asterisk is the default type, which can be changed by selecting its name in the list, and clicking on the Set Default button.

There is a limit of 245 user-defined types in each data file.

If the radio button "step matrices" at the upper right is selected, then the list of types will include all available types, even the predefined types and the character state tree types. For any selected type (except Dollo) the matrix of state-to-state distances of types is shown. Even though the character state tree types are not defined in terms of matrices, a matrix is shown for these to show you what the character state tree implies about state-to-state distances. (However, character state tree types cannot be edited in matrix form.) If the radio button "state trees" at the upper right is selected, then the list includes only the user-defined character state tree types and the predefined types that can be represented as a character state tree. The only predefined type shown is ordered.

If you have a choice of representing a user-defined type as a character state tree or a matrix, choose the character state tree: not only is it much easier to edit, but other programs such as PAUP* are much more efficient at dealing with character state trees than with step matrices. MacClade 4 internally treats the two identically (both slowly).

To create a new type, click on the New button. A dialog box will be presented, allowing you to name the type. Type names can be at most 31 characters long; if you type more, MacClade will truncate the name to 31 characters. The new type will be of the matrix or character state tree sort, depending upon whether the dialog box was displaying matrices or trees when the type was created.

You can copy the specifications of a type into a private temporary clipboard with the Copy button. If you then create or select a second type, and then press the Paste button, the specifications of the first type will be pasted into the second type, replacing its previous specifications.

If you press the Apply button, then any currently selected characters in the frontmost window below the **Type Edit** dialog box will be set to the type currently displayed in the **Type Edit** dialog box.

If you press OK at the bottom right of the **Type Edit** dialog box, then all editing of types will be accepted and stored in memory; if you press Cancel, they will be forgotten. There is one exception. If you delete user-defined types that are in use or that were used in any stored type set, MacClade will set the relevant characters to be of default type. Pressing Cancel after such a deletion will not reset the types of characters to be the resurrected user-defined type; they will remain of default type.

Step-matrix types

To edit a step-matrix type, select one or more elements in the matrix, and type a new number into the box above Enter to indicate how many steps are counted for a change from one state to the other. You can select multiple elements by holding down the Shift key as you continue to select. When you click on the Enter button (or enter key), MacClade will replace the selected matrix elements with the entered number.

The number of steps you indicate can be either integral (0 to 999) or with decimal points (0.0 to 99.9). Typing uppercase or lowercase i, or the infinity symbol ∞ , will signify an impossible transition.

A matrix must be entirely integral or entirely decimal. To convert a matrix from integral to decimal or vice versa, use the Integral/Decimal buttons. MacClade will give you a choice. On converting decimal to integral, MacClade asks if you want to truncate to the next lowest integer (e.g., 9.1 becomes 9) or to multiply by 10 first (9.1 becomes 91). The second option converts matrices to integral without losing accuracy. For integral to decimal conversions, MacClade asks if you want to append a decimal point (e.g., 91 becomes 91.0) or to divide by 10 first (91 becomes 9.1).

The matrix has to be big enough to include all states in the characters to which it is applied. That is, if you have a step matrix that indicates the costs for transformations among states 0, 1, and 2 only, MacClade will

not allow you to apply it to a character with states 0, 1, 2, 3, and 4. Thus the matrix must be large enough to accommodate the characters. The matrix can be unnecessarily large, in that you can use a matrix that defines costs for transformations among states 0 through 9 even if you apply it to a character with only 3 states. However, using an unnecessarily large matrix will slow down calculations. It is therefore best to adjust the size of the step matrix to fit the characters to which it will be applied. To make a bigger or smaller matrix, touch on the button Change Size and in the dialog box presented type in the symbol of the maximum state.

You can copy and paste already-defined matrices from other types, except that you cannot copy the matrices for Dollo and stratigraphic types.

You cannot edit the matrix if the type shown is predefined or a character state tree type, but you can copy such a matrix and paste it into a new matrix type. The one exception to the rule that predefined types cannot be edited is the stratigraphic type: you can edit this type to a limited extent. The stratigraphic type matrix allows you to alter the cost of an absence in each of the strata.

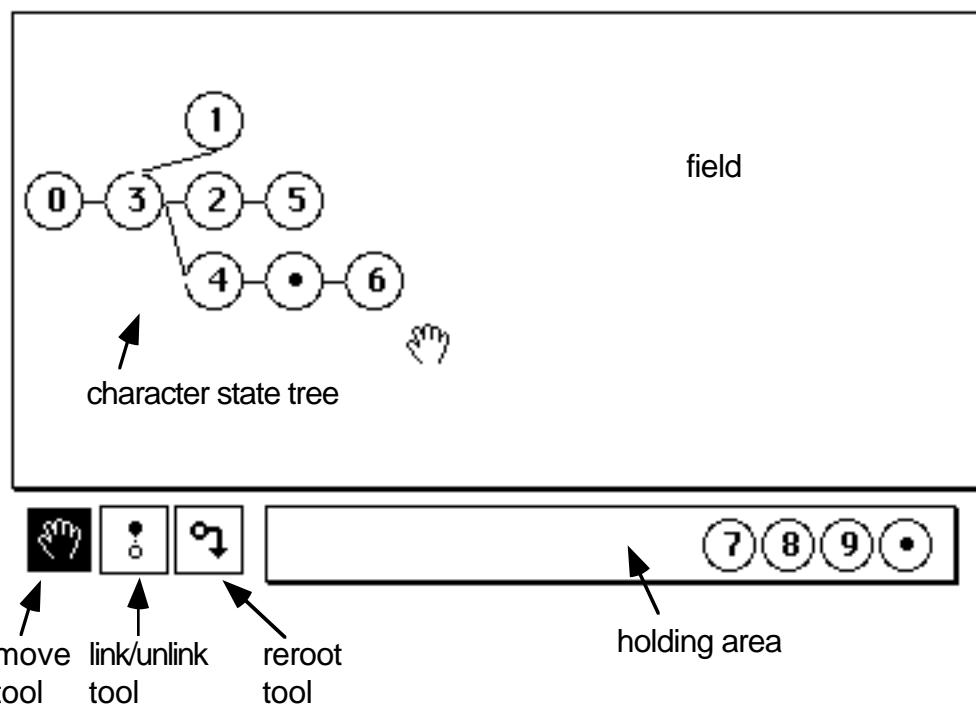
To make a matrix symmetrical, click on the Symmetrize button. You will have three options: copy the lower-left triangle of the matrix to the upper-right triangle; copy the upper-right triangle to the lower-left; or averaging. If averaging is chosen, MacClade will place as the number of steps from i to j the average value of the number of steps from i to j and j to i . If the type matrix originally consisted of integral values, you may wish to ask MacClade first to convert it to decimal values for greater accuracy.

The triangle inequality

MacClade does not prohibit types that violate the triangle inequality (["Violating the triangle inequality" on page 72](#)), but because violations may be unintentional and can have rather strange consequences on the reconstruction of character evolution, MacClade checks for violations of the triangle inequality when a file is read, and reports any to the user. Also, in the **Type Edit** dialog box, you can ask MacClade to check types for violation using the **Check Δ** button. If you wish, MacClade will automatically fix each violating type by setting the cost of going between any two states to be the minimum cost along any path between the two states. (One might imagine that this procedure would give different results depending upon the sequence in which the various costs in the matrix are adjusted. However, empirical studies on 2,000 random cost matrices violating the triangle inequality suggest that this is not the case [D. Maddison, unpublished].)

Character state tree types

Character state tree types appear in the **Type Edit** dialog box as dots containing state symbols, linked by lines (see figure, below). Under the large rectangular field containing the character state tree is a holding area for states not yet incorporated into the tree. There are three tools available for rearranging the tree: a hand tool for moving state dots around, a link/unlink tool for connecting and disconnecting them, and a reroot tool for choosing a new root state. Move the tool down to the row of circles with state symbols. Pick up a state dot and drop it into the large rectangular field. Go back down and pick up another state. Join the two states by clicking on one in the field, dragging, and dropping the line on the other state in the field. Disconnect states by repeating this using the link/unlink tool. Unnamed, unobserved states can be added by using the "•" state. Unnamed states cannot be left in terminal position in the tree; MacClade will warn you and remove these states.



You cannot create a loop or circuit in a character state tree. Also, all states in the field must be connected to one another before you click on Clean Up or move to another type. (If not, MacClade will remove those not connected to the root.) The first state placed in the field is drawn as the root of the character state tree. You can reroot the tree by touching on a new root with the reroot tool. The root has aesthetic value only, as MacClade 4's character state trees are undirected (effectively unrooted); root state is not implied to be ancestral, nor is it implied that evolution occurs only from the root to the tips. Character state tree types are fully reversible. In the tree figured above, evolution can proceed from state 4 to 3 to 1, for example. If you click on the Clean Up button, the character state tree will be redrawn in an evenly spaced fashion. If you originally define a type as a character state tree, you can see its matrix form by pressing on the "step matrix" radio button, but you cannot edit the matrix directly.

WARNING: MacClade does not allow the unnamed, unobserved states, represented by spots, to be reconstructed as ancestral states. They are considered only as indications of how distant the observed states (e.g., A, C, G, T) are from one another. PAUP* does not follow this convention, and instead allows the unobserved states to be reconstructed as ancestral states where this would allow a parsimonious reconstruction. This means that whenever a character state tree has unnamed, unobserved states in it, PAUP* and MacClade may differ in the treelengths, reconstructions of character evolution, and other results that they report.

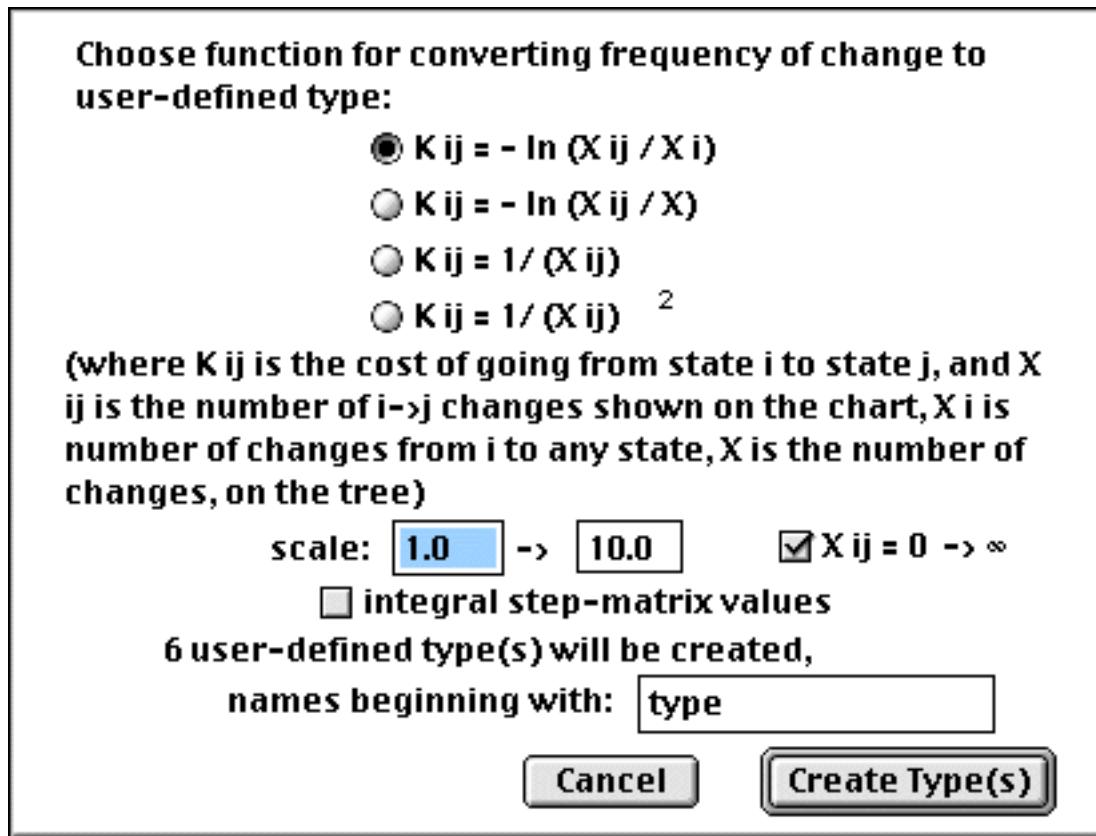
Creating protein parsimony types

For protein data, MacClade will create a user-defined type based on the minimum number of amino acid substitutions required by the current genetic code. For details, see ["Creating protein-parsimony user-defined types" on page 309](#).

Creation of a transformation type based on reconstructed frequency of changes

You can create a transformation type that is a function of the reconstructed frequency of changes in all or some of the characters, as calculated by the Changes & Stasis chart (["State Changes & Stasis" on page 396](#)).

Choosing **Chart to Type** from the **Chart** menu presents you with the following dialog box in which you choose options to convert a State Changes & Stasis chart to a transformation type.



In general, these four functions are of this form: Cost of an i to j change in the step or cost matrix is equal to a function of the inferred frequency of that change on the tree(s). The four functions are all monotone decreasing; that is, the higher the observed frequency, the lower the cost. The first two functions provided are similar to those proposed by W. Wheeler (1990); the latter two are those suggested by Williams and Fitch (1989, 1990).

You can set the scale of the values in the user-defined type that will be created. By default, the costs are scaled to fit in to the range 1 to 10, so that the least frequent kind of change will be given a weight of 10.0, and the most frequent a cost of 1.0. If you set " $X_{ij} = 0 \rightarrow \infty$ ", then any change with zero frequency will be set to a cost of infinity (that is, it will be presumed impossible); otherwise, it will be set to the maximum value of the range. If "integral step-matrix values" is chosen, then the values in the step matrix will be rounded to the nearest integer. If you enter a value that is above the maximum allowable by MacClade, the program will beep.

You can name the type to be created in the box provided. If there is a range of values for an i to j change, then MacClade will produce more than one user-defined type. For example, if there were minimum, average, and maximum values of each change, then MacClade would produce three types, by default named "type min", "type avg", and "type max".

These types can then be applied to the characters, new trees found, new types created based on the frequency of changes on those new trees, and so on. This is a variant of Williams and Fitch's (1989, 1990) method of Dynamic Weighting. Dynamic Weighting simultaneously infers a probabilistic model and a tree using an iterative process. A step matrix is derived from an initial estimate of relative frequencies of change, a tree is then inferred using parsimony, new estimates of relative frequencies of change are made

based on that tree, from these a new step matrix is formed, a tree is inferred assuming this step matrix, and so on, until the estimate of the step matrix in one iteration is the same as in the next. This method has been justified primarily by empirical studies (Fitch and Ye, 1992). MacClade's algorithms differ in that node numbering is not performed, and that equally parsimonious reconstructions of character evolution are weighted equally in calculating the weights based on average number of changes over all reconstructions. Rather than using node numbering, we recommend asking MacClade to fix violations of the triangle inequality (see "[Step Matrices" on page 72](#)) if you want to avoid the triangle inequality. Some aspects of Dynamic Weighting are not well defined. For example, should one use the user-defined type created from the minimum values across multiple reconstructions and trees, the average values, or the maximum values? We do not necessarily advocate use of Dynamic Weighting, but we have placed this capability in MacClade to allow you to explore the behavior of the method.

Importing types from other files

The **Type Import** menu item in the **Characters** menu allows you to load in types from other NEXUS files. If, during the process of loading types from another file, MacClade encounters duplicate type names, it will ask you how to resolve the conflict. MacClade assumes that the other file will be using the same state symbols. If it is not, problems may arise, and MacClade will give a warning and abandon importing the types.

Type sets

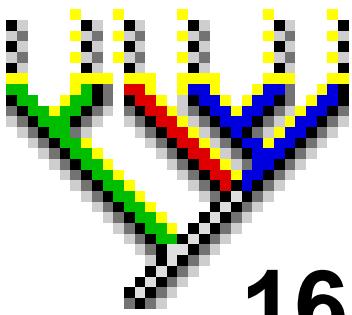
Type sets can be stored, retrieved, renamed, and deleted in the same fashion as weight sets ([page 275](#)), using the **Type Sets** submenu items and the type sets list window.

Limitations on the use of various character types

Some character types cannot be used with some of the features of MacClade. For instance, with a polytymous tree no included characters can be of user-defined types. The following table indicates which features are available with which character types. Because of these limitations, you will notice that some options will be occasionally unavailable to you (for instance, you cannot change a character to a Dollo type if the current tree has polytomies), or that MacClade will give a warning when you try to use them.

| Feature | Unordered | Ordered | Irreversible | Stratigraphic | Dollo | User-defined |
|----------------------------------|-----------|---------|--------------|---------------|-------|--------------|
| Treelength | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| Trace Character | | | | | | |
| Branch swapping | | | | | | |
| All MPRs | | | | | | |
| Tree changes | ✓ | ✓ | ✓ | | ✓ | ✓ |
| Trace All Changes | | | | | | |
| Branch length | | | | | | |
| Consistency index shown | ✓ | ✓ | ✓ | | | |
| Retention index | ✓ | ✓ | | | | |
| Rescaled consistency index | | | | | | |
| Polytomies | ✓ | ✓ | | | | |
| Observed taxa fixed as ancestors | ✓ | ✓ | ✓ | ✓ | | |
| DELTRAN/ ACCTRAN | ✓ | ✓ | | | | |

The only two fully supported types are unordered and ordered. Most limitations are a result of our not yet having implemented the feature for a particular character type. Users should consult the descriptions of each of these features for information on why certain character types cannot be handled.



PREPARING MOLECULAR DATA

16

MacClade was designed for use with morphological, behavioral, molecular, or other classes of character data. In this chapter we describe some of the special features in MacClade for use in preparing molecular data.

Importing data from molecular databases

Molecular databases normally supply data in formats such as NBRF, GenBank, GCG/MSF, and so on. For information about importing matrices in these formats, see [Chapter 8](#).

If you wish to import sequences into an already existing file, see ["Importing sequences into an existing file" on page 147](#).

Working with alignment programs

MacClade has several features for use in conjunction with sequence alignment programs. MacClade can export files in formats that alignment programs can read (e.g., NBRF; see [page 144](#)), and it can import files in formats that alignment programs generate (e.g., Clustal's "aln" format; see [page 145](#)). In addition, the ability to maintain long taxon names through the bottleneck of some alignment programs (which may truncate taxon names) is valuable (see ["Condensing and resurrecting taxon names" on page 183](#)).

DNA, RNA, and protein data formats

Three of MacClade's data formats (DNA, RNA, and protein; see ["Data formats" on page 192](#)) come pre-formatted with appropriate symbols defined (e.g., "A", "C", "G", and "T"). The symbols used are the standard IUPAC symbols (see ["Symbols for states" on page 195](#)).

Some features in MacClade are available only if DNA or RNA formats are chosen (e.g., tabulations of transitions and transversions in some charts, translation of nucleotide sequence to amino acid sequence, and so on).

Specifying coding regions and codon positions

For DNA or RNA sequence data, sites are specified as to whether or not they are in protein-coding regions. Sites in coding regions are designated as being either in the first, second or third position of a codon. You can assign such positions manually, or ask MacClade to calculate them automatically. The coding status and position of a site is shown in the Pos column in the character list window. The symbol "-" indicates a non-coding site.

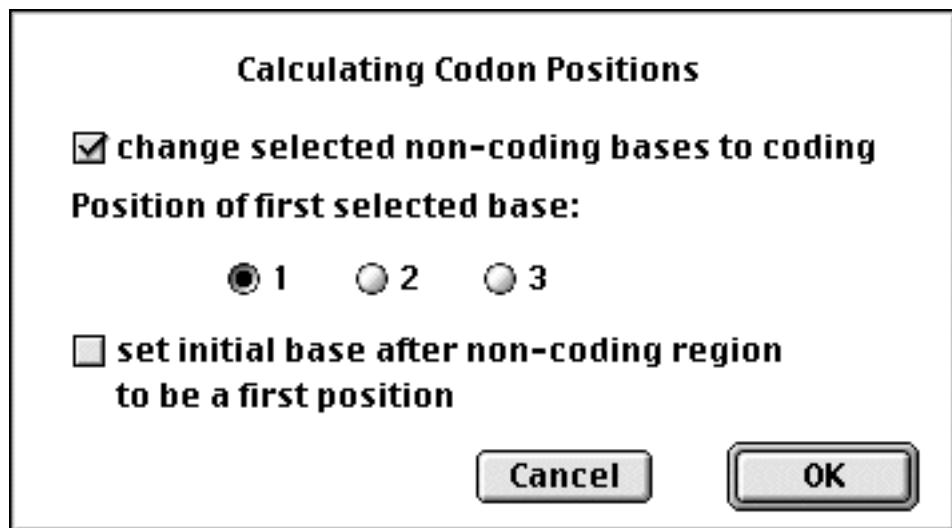
When you start a new DNA or RNA data file, or add new sites to an existing matrix, MacClade assumes the new sites are non-coding. You can designate sites as coding either by indicating their codon positions as first, second or third, or by asking MacClade to calculate codon positions automatically. To set a site manually as first, second, or third position, or non-coding, select it in the character list window or data editor and choose the appropriate menu item in the **Codon Positions** submenu.

To have MacClade calculate positions automatically, select a contiguous block of characters in either the data editor or in the character list window ([page 163](#)), as shown in the following example:

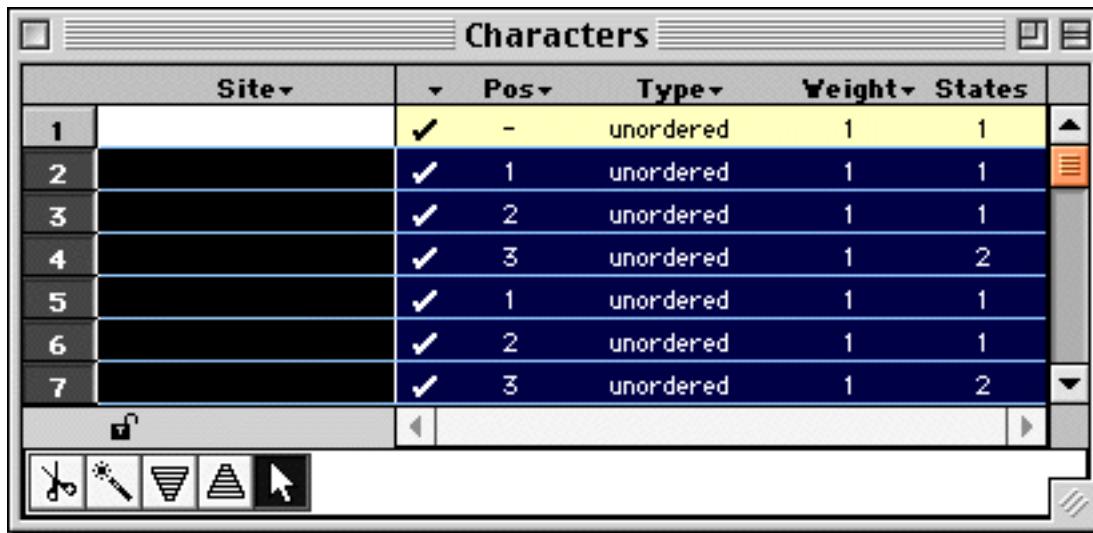
The screenshot shows the 'Characters' dialog box in MacClade. The table has columns for Site, Pos, Type, Weight, and States. Rows 1 through 7 are selected, indicated by yellow highlighting. The 'Type' column for all selected rows is set to 'unordered'. The 'Weight' column shows values 1, 1, 1, 1, 2, 1, 1 respectively. The 'States' column shows values 1, 1, 1, 1, 2, 1, 2 respectively. The bottom toolbar contains icons for selection, zoom, and other data manipulation.

| Site | Pos | Type | Weight | States |
|------|-----|-----------|--------|--------|
| 1 | - | unordered | 1 | 1 |
| 2 | - | unordered | 1 | 1 |
| 3 | - | unordered | 1 | 1 |
| 4 | - | unordered | 1 | 2 |
| 5 | - | unordered | 1 | 1 |
| 6 | - | unordered | 1 | 1 |
| 7 | - | unordered | 1 | 2 |

Then choose **Calculate Positions** from the **Codon Positions** submenu of the **Characters** menu. In the dialog box, choose whether or not you want MacClade to reset bases currently designated as non-coding to be coding, and then set the position of the first selected base.



If you then press Calculate, MacClade will automatically assign bases to first, second, and third positions:



If the first base in the selected block is part of a coding region, then MacClade will assign to it the position number you have requested. For example, if the first site is designated a second position and is in a coding region, then MacClade will set the positions to 2, 3, 1, 2, 3, 1, 2, 3, ... and so on. If you so choose, the first base past a non-coding block will be designated a first position:

| | |
|---------------------|---------------------------|
| sequence | ACCCGACGACATGACAGTACAGTTA |
| coding/non-coding | ccccccccc-----ccccccccc |
| calculated position | 3123123-----1231231231 |
| assigned position | 3123123-----1231231231 |

Otherwise, MacClade will continue the numbering through the non-coding block:

| | |
|---------------------|---------------------------|
| sequence | ACCCGACGACATGACAGTACAGTTA |
| coding/non-coding | ccccccccc-----ccccccccc |
| calculated position | 3123123123123123123123 |
| assigned position | 3123123-----3123123123 |

The purpose of the codon position designation is for display and selection in the character list window, for DNA/RNA-to-protein translation, and for various charting options. The **Select** submenu in the **Edit** menu, available if the character list window is frontmost, will contain, for example, an **Every 2nd Position** menu item, the selection of which will cause every second position in the character list window to be selected. This allows you to exclude all second and third codon positions to focus on the evolution of nucleotides in the first codon position, or to give all third-position nucleotides low weight in a phylogenetic analysis, and so on. Codon position can be used in the chart window. For instance, the **By Codon Position** option in the Character Steps/etc. chart can indicate the number of steps in nucleotides at the first, second and third positions (see [page 392](#)). Codon position designations are important for DNA/RNA-to-protein translation, as they specify the codons that get translated to amino acids.

If you have designated codon positions, the character number will be displayed in varied colors in the data editor: blue, green, and red for the first, second, and third codon positions, respectively (see "[Coloring codon positions](#)" on page 268):

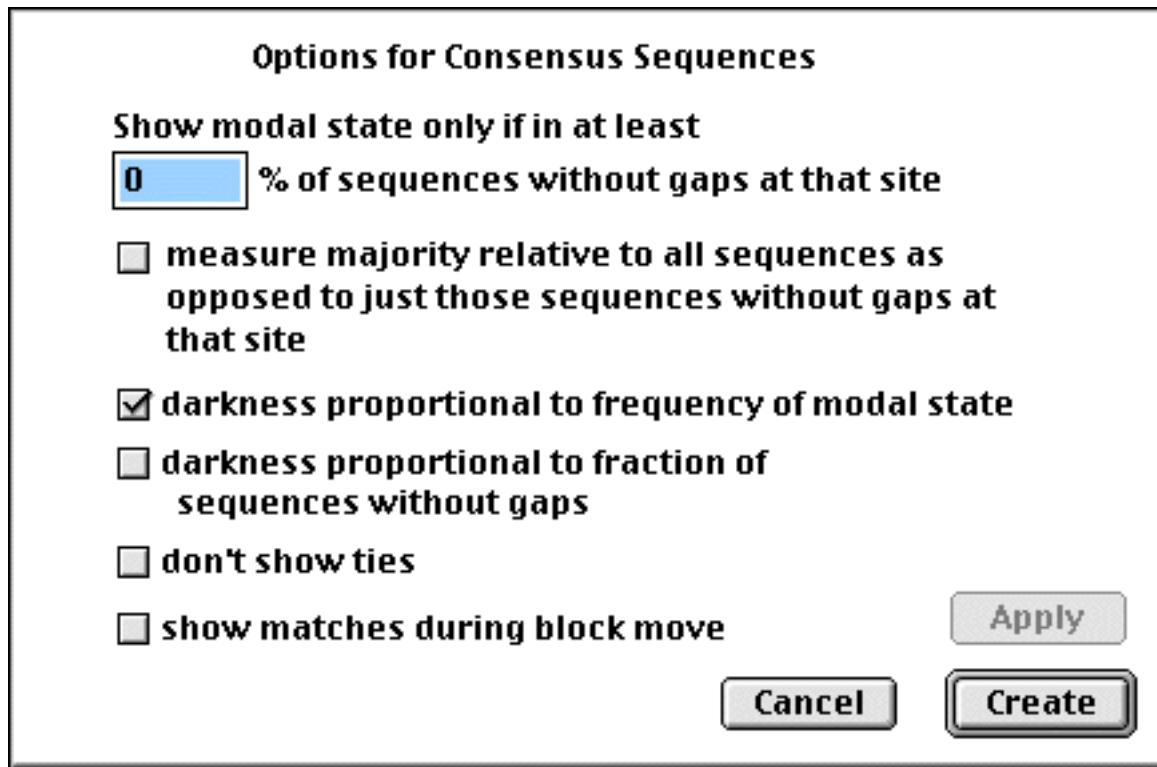
| | | Characters | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 |
|---|--------------|------------|---|---|---|---|---|---|---|---|---|----|----|----|----|----|----|----|----|----|----|
| | | Taxa | | | | | | | | | | | | | | | | | | | |
| 1 | Homo sapiens | A | A | G | C | T | T | C | A | C | C | G | G | C | G | C | A | G | T | C | |
| 2 | Pan | A | A | G | C | T | T | C | A | C | C | G | G | C | G | C | A | A | T | T | |

Consensus sequences

Consensus sequences can be displayed in a data matrix, and summarize the modal state for each character:

| Modal | | A | A | G | C | T | T | C | A | C | C | G | G | C | G | C | A | A | Y | C |
|-------|----------------|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | Homo sapiens | A | A | G | C | T | T | C | A | C | C | G | G | C | G | C | A | G | T | C |
| 2 | Pan | A | A | G | C | T | T | C | A | C | C | G | G | C | G | C | A | A | T | T |
| 3 | Gorilla | A | A | G | C | T | T | C | A | C | C | G | G | C | G | C | A | G | T | G |
| 4 | Pongo | A | A | G | C | T | T | C | A | C | C | G | G | C | G | C | A | A | C | A |
| 5 | Hylobates | A | A | G | C | T | T | T | A | C | A | G | G | T | G | C | A | A | C | G |
| 6 | Macaca fuscata | A | A | G | C | T | T | T | T | C | C | G | G | C | G | C | A | A | C | A |

To create a consensus sequence, choose **Consensus Sequence** from the **Display** menu. This dialog box will appear:



Pressing Create will create the consensus sequence. It will be shown just below the character names.

MacClade calculates the most frequent (modal) state or states, but displays them only if they meet the frequency specified. In particular, if you put 50% in the "Show modal state only if in at least ___ % of sequences without gaps at that site", then if a state appears in only 40% of the sequences, it would not be shown even if it is the modal state. The percentage is measured by default only over those sequences with nongaps at that site, but if you check "measure majority relative to all sequences..." then the percentage would be measured over all sequences. For example, for the following site,



G is present in 75% of the sequences with nongaps, but only 50% of all sequences.

By default, the modal state is dimmed in proportion to its frequency, with modal states that are present in a low percentage of sequences being pale. You can turn this feature off by unchecking "darkness proportional to frequency of modal state", in which case modal states will not be dimmed if they are in a low frequency. You can also ask the modal states to be dimmed if there are few sequences with data at that site, by checking "darkness proportional to fraction of sequences without gaps".

If you check "don't show ties", then if there are two or more modal states of equal frequency, they will not be shown.

When you use the Block Mover ([page 242](#)) on a piece, and if you choose "show matches during block move" in the Consensus Sequences dialog box, the box with the name of the consensus sequence changes to have in it the total number of elements in the consensus sequence, as shown below:

| | | Characters | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 |
|------|--------------|------------|---|---|---|---|---|---|---|---|---|----|----|----|----|----|
| Taxa | | | | | | | | | | | | | | | | |
| | 85/83 | | | | C | T | | T | T | | G | G | | T | C | |
| 1 | Thylacinus* | - | - | - | - | C | T | T | T | G | G | A | T | C | C | |
| 2 | Sarcophilus* | - | - | C | T | | G | G | G | T | C | T | T | T | T | |

The first number is the total number of elements in the consensus sequence; the second number is the total number that would be there if the "measure majority relative to all sequences..." option were activated for that consensus sequence.

You can add as many consensus sequences as you wish, and have them of different stringencies:

| Taxa | Characters | 1 | 31 | 41 | 51 | 61 | 71 | 81 | 92 | 02 | 12 | 22 | 32 | 42 | 52 | 62 | 72 | 82 | 93 | 03 | 13 | 23 | 33 | 43 |
|---------------------------|---------------------|---|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| Modal | | A | C | T | A | G | G | A | A | T | C | T | G | C | C | T | A | A | T | T | A | T | T | |
| Modal, ≥75% | | A | | T | A | G | G | A | A | T | | T | G | C | C | T | A | A | T | A | T | | | |
| Modal, ≥90%, ties omitted | | | | T | | G | G | | T | | T | G | C | T | A | | | T | | | | | T | |
| 1 | Thylacinus* | A | C | T | A | G | G | A | A | T | C | T | G | C | C | T | A | G | T | C | A | T | T | |
| 2 | Sarcophilus* | A | T | T | A | G | G | A | A | T | A | T | G | C | C | T | A | A | T | T | A | T | T | |
| 3 | Dasyurus* | A | T | T | A | G | G | A | G | T | A | T | G | C | C | T | A | A | T | T | A | T | T | |
| 4 | Echymipera* | A | C | T | A | G | G | A | A | T | C | T | G | C | T | T | A | G | T | C | A | T | C | |
| 5 | Trichosurus* | A | C | T | A | G | G | C | A | T | C | T | G | C | T | T | A | A | C | T | A | T | A | |
| 6 | Phalanger* | A | T | T | A | G | G | T | T | T | C | T | G | C | C | T | A | A | C | A | A | T | C | |
| 7 | Philander | T | C | T | A | G | G | A | A | T | A | T | G | C | C | T | A | A | T | T | A | T | T | |
| 8 | Bos | C | C | T | G | G | G | A | A | T | C | T | G | C | C | T | A | A | T | C | C | T | A | |
| 9 | | | | | | | | | | | | | | | | | | | | | | | | |

Once added, you can reorder them by dragging, and you can delete them by selecting them and hitting the Delete key. If you double-click on a consensus sequence, the Consensus Sequences dialog box will appear, in which you can modify the settings for that consensus sequence and then press Apply to invoke those changes.

Consensus sequences are updated automatically as you edit the matrix. They are not used in tree calculations. However, you can move a consensus sequence to be a true member of the matrix by selecting it and choosing **Transfer Consensus to Taxon** from the **Taxa** menu (see [page 181](#)).

If you choose a taxon set in the **Consensus Taxa** submenu of the **Display** menu, then the consensus sequences will be calculated only over elements of that taxon set. By default the consensus is calculated over all taxa. You must first create other taxon sets to use them in this way (do this by selecting rows in the taxon list window, and choosing **Store Taxon Set** from the **Taxon Sets** submenu of the **Taxa** menu; see [page 189](#)).

Alignment of molecular sequences within MacClade

MacClade 4 has many features that aid in manual alignment of molecular sequences. Many of these are documented fully in [Chapter 13](#) (especially the section "[Moving data cells & sequence alignment](#)" on [page 240](#)) and [Chapter 14](#); see also the section "[Calculating the number of stop codons in protein-coding nucleotide sequences](#)" on [page 381](#)). Here we provide an example of use of some of the tools for one example, the "Insect DNA" file in the Molecular folder of the Examples folder.



The "Insect DNA" matrix has protein-coding DNA sequence; note that the cells are colored to match that of the amino acid to which they would be translated (see "[Coloring cells](#)" on [page 259](#)). A quick glance at the sequences reveals that one of them appears misaligned:



| | |
|----------|---|
| insect 1 | CACGGCATGTCTGGCTCCTGCACTGTAAAAACCTGCTGGATGCGACTT |
| insect 2 | CATGGTATGTCTGGCTCCTGCACTGTAAAAACCTGCTGGATGCGTCTT |
| insect 3 | CACGGTATGTCTGGCTCCTGCAACCGTCAAAACCTGCTGGATGCGCTC |
| insect 4 | CACGGTATGTCTGGCTCCTGCAACCGTAAAAACCTGCTGGATGCGACTT |
| insect 5 | -----TGTACTGTCAAAACGTGGATGCGTCTTCCDAACTTCCGTGT |

To align the last sequence, use the Block Mover (; see [page 247](#)) to grab the last sequence and push it to the right, until the first few bases match well with the other sequences:

| | |
|----------|---|
| insect 1 | CACGGCATGTCTGGCTCCTGCACTGTAAAAACCTGCTGGATGCGACTT |
| insect 2 | CATGGTATGTCTGGCTCCTGCACTGTAAAAACCTGCTGGATGCGTCTT |
| insect 3 | CACGGTATGTCTGGCTCCTGCAACCGTCAAAACCTGCTGGATGCGCTC |
| insect 4 | CACGGTATGTCTGGCTCCTGCAACCGTAAAAACCTGCTGGATGCGACTT |
| insect 5 | -----TGTACTGTCAAAACGTGGATGCGTCTTCCC |

It appears as if there was a deletion of one codon in the last sequence (or, an insertion in the ancestor of the other sequences), and gaps need to be introduced in it to improve the alignment. To do this, use the Block Splitter (; see [page 244](#)),



| | |
|----------|---|
| insect 1 | CACGGCATGTCTGGCTCCTGCACTGTAAAAACCTGCTGGATGCGACTT |
| insect 2 | CATGGTATGTCTGGCTCCTGCACTGTAAAAACCTGCTGGATGCGTCTT |
| insect 3 | CACGGTATGTCTGGCTCCTGCAACCGTCAAAACCTGCTGGATGCGCTC |
| insect 4 | CACGGTATGTCTGGCTCCTGCAACCGTAAAAACCTGCTGGATGCGACTT |
| insect 5 | -----TGTACTGTCAAAACGTGGATGCGTCTTCCC |

and open up three gaps in front of :

| | |
|----------|---|
| insect 1 | CACGGCATGTCTGGCTCCTGCACTGTAAAAACCTGCTGGATGCGACTT |
| insect 2 | CATGGTATGTCTGGCTCCTGCACTGTAAAAACCTGCTGGATGCGTCTT |
| insect 3 | CACGGTATGTCTGGCTCCTGCAACCGTCAAAACCTGCTGGATGCGCTC |
| insect 4 | CACGGTATGTCTGGCTCCTGCAACCGTAAAAACCTGCTGGATGCGACTT |
| insect 5 | -----TGTACTGTCAAAACGT---TGGATGCGTCTT |

That's a bit better. Now, let's import a sequence into this matrix. Use the **NBRF File** item of the **Import Sequences** submenu (see [page 147](#)) of the **Taxa** menu, press Import in the dialog box that appears, and open the file "new insect.NBRF" in the Molecular folder. This should add the new sequence to the end of the matrix:

| | |
|----------|---|
| insect 1 | CACGGCATGTCTGGCTCCTGCACTGTAAAAACCTGCTGGATGCGACTT |
| insect 2 | CATGGTATGTCTGGCTCCTGCACTGTAAAAACCTGCTGGATGCGTCTT |
| insect 3 | CACGGTATGTCTGGCTCCTGCAACCGTCAAAACCTGCTGGATGCGCCTC |
| insect 4 | CACGGTATGTCTGGCTCCTGCAACCGTAAAAACCTGCTGGATGCGACTT |
| insect 5 | -----TGTACTGTCAAAACG-----TGGATGCGTCTT |
| new | GTACTGTACCGTGAAAGACTTGTTGGATGCGTCTTCCAAAGTTCCGAGT |



Rather than manually aligning this sequence to the remainder, let's use the pairwise alignment tool (; see [page 242](#)) to align the new sequence to the "insect 5" sequence. To do this, touch on the new sequence with the pairwise alignment tool, and drag the new sequence onto the insect 5 sequence. This will then align "new" against "insect 5":

| | |
|----------|--|
| insect 1 | CACGGCATGTCTGGCTCCTGCACTGTAAAAACCTGCTGGATGCGACTT |
| insect 2 | CATGGTATGTCTGGCTCCTGCACTGTAAAAACCTGCTGGATGCGTCTT |
| insect 3 | CACGGTATGTCTGGCTCCTGCAACCGTCAAAACCTGCTGGATGCGCCTC |
| insect 4 | CACGGTATGTCTGGCTCCTGCAACCGTAAAAACCTGCTGGATGCGACTT |
| insect 5 | -----TGTACTGTCAAAACG-----TGGATGCGTCTT |
| new | -----GTACTGTACCGTGAAAGACTTGTTGGATGCGTCTTCCAAAGTTCCGAGT |

Aligning nucleotides to match a protein alignment

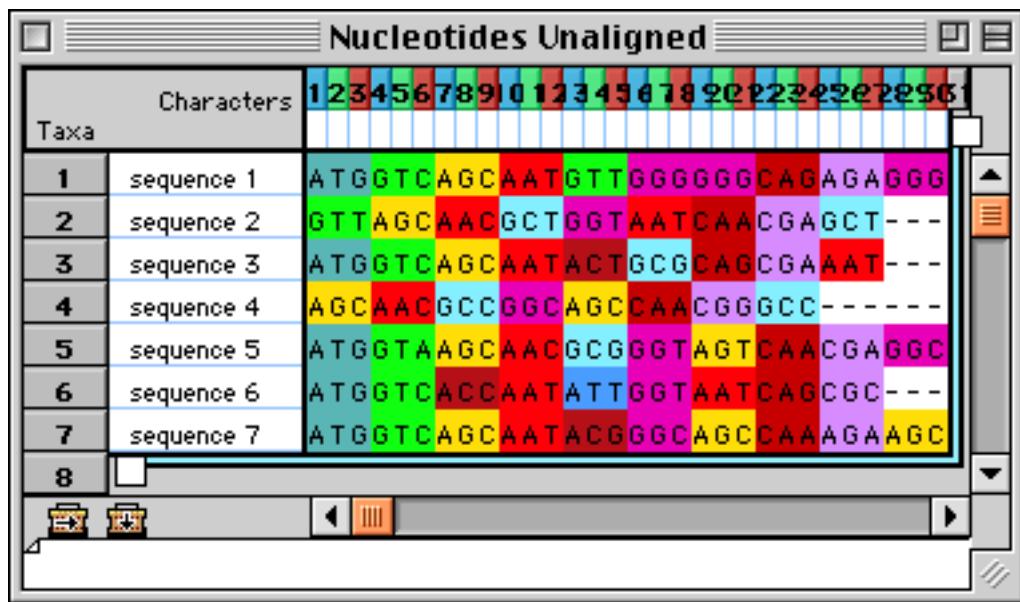
You may wish to align your nucleotides to match an existing alignment of amino acids. MacClade has a limited capacity to do this, it does not conduct extensive error-checking, and there are restrictions. In particular:

1. The protein alignment must be stored in an NBRF file.
2. There must be an exact 3:1 match between the nucleotides present in the DNA/RNA file and the amino acids present in the protein file. There can be no partial triplets in the DNA/RNA file, no nucleotides present in the DNA/RNA file that are not represented by amino acids in the protein file, and no amino acids in the protein file not represented by nucleotides in the DNA/RNA file.
3. The taxon names in the two files must be identical (although they need not be in identical orders).

If these conditions hold, then you can ask MacClade to align the nucleotides in an open file to match those in an external NBRF protein alignment. To do this, have the nucleotide file open in MacClade's data editor. Then, hold down the Option key, and choose **Import NBRF Protein Alignment** from the **Utilities** menu. In the window presented, choose the NBRF file containing the protein alignment. MacClade will then realign the nucleotides to match this protein alignment.



For example, the file "Nucleotides Unaligned" in the Miscellaneous folder contains a small piece of protein-coding DNA for several sequences that are unaligned:



The file "Protein Alignment.nbrf" contains an alignment of the translated amino acids for these same data:

| | |
|------------|---------------------|
| sequence 1 | M V S N V G G Q R G |
| sequence 2 | - V S N A G N Q R A |
| sequence 3 | M V S N T - A Q R N |
| sequence 4 | - - S N A G S Q R A |
| sequence 5 | M V S N A G S Q R G |
| sequence 6 | M V T N I G N Q R - |
| sequence 7 | M V S N T G S Q R S |

With the "Nucleotides Unaligned" matrix displayed in MacClade's data editor, choosing **Import NBRF Protein Alignment** from the **Utilities** menu, and then opening the "Protein Alignment.nbrf" file, will cause MacClade to realign the nucleotides as shown below:

Other data editor features for sequence data

Data display

MacClade's data editor has a number of display options specifically designed for molecular data. For example, data cells can be colored in various ways (see ["Coloring cells" on page 259](#)). The amino acids to which nucleotide triplets would be translated can be displayed using these colors or using the **Show Translated AAs** feature (see ["Amino acid translation" on page 261](#)). Character sets can be shaded, which can be useful to distinguish introns or other regions of note (see ["Shading character sets" on page 266](#)). Codon positions can be indicated in the editor with colors (see ["Coloring codon positions" on page 268](#)). Full details of these and other features are given in [Chapter 14](#).

Match first

This matrix display option, described in ["Matching first taxon" on page 258](#), uses the first taxon as a standard and displays cells in the rest of the matrix with a chosen symbol (for instance, ".") whenever the state matches that in the first taxon.

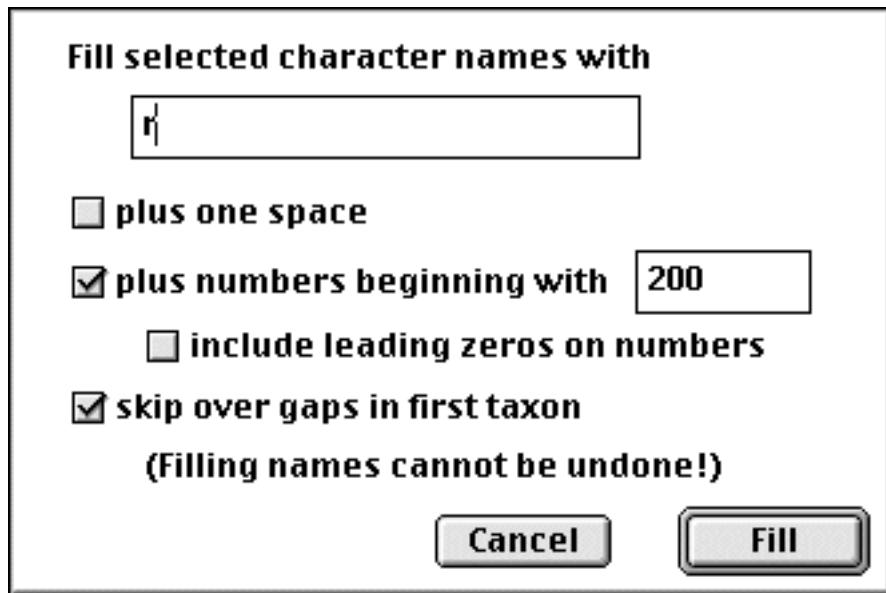
If "match first mode" is on in the data editor, you can type the matching symbol into any data cell and MacClade will interpret that to mean you are entering the same state as in the first taxon. This, of course, does not work for entering data for the first taxon.

Naming sites

You may wish to provide character names to the sites along the sequence, so that you can track the sites even as you delete or add new sites. MacClade's **Fill** command (**Utilities** menu) allows rapid entry of sequentially numbered character names.

For example, imagine that you have an aligned set of sequences, 450 sites in length, one sequence of which

corresponds to sites 200–600 of some reference sequence (with 50 gaps added into the reference sequence during the alignment). To assign character names based on this reference sequence, move that reference sequence to be taxon 1, and select the full set of character names. The easiest way to do this would be to touch on the character name of the first site, then hold down the Shift and Option keys and press the right arrow (\rightarrow) key (down arrow [\downarrow] key if the matrix is transposed), or to use the Selection Wand (page 213). You are now ready to call up the **Fill** dialog box, and set it up as shown below.



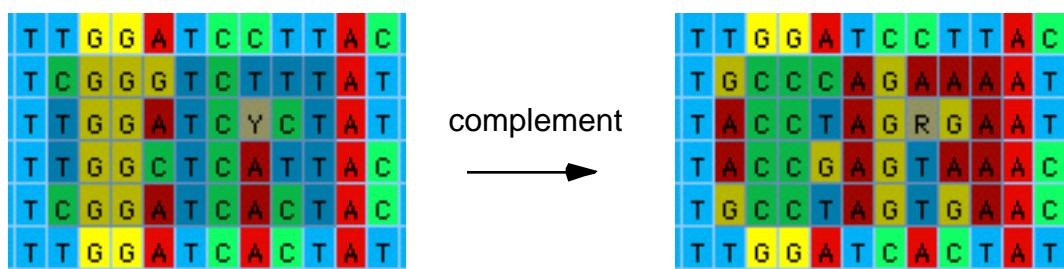
With "skip over gaps in first taxon" selected, MacClade will number the sites based on the reference sequence:

| | | | | | | | | | | |
|-----------|---|---|---|---|---|---|---|---|---|---|
| reference | A | - | - | C | - | G | T | G | C | T |
| sequence2 | A | G | G | C | - | G | T | G | C | T |
| sequence3 | A | - | - | C | C | G | T | G | C | T |

r200 r200^a r200b r201 r201^a r202 r203 r204 r205 r206

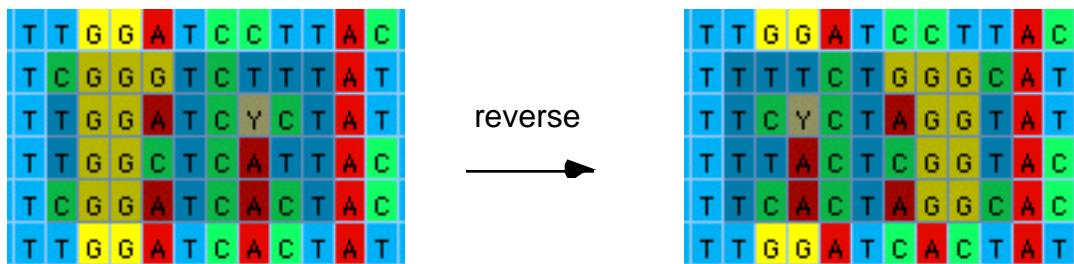
Changing nucleotides to their complements

To convert all nucleotides in a selected block of cells to their complements, choose **Complement** from the **Utilities** menu. This will convert all A→T, all T→A, all G→C, and all C→G, as illustrated in the example below:



Reversal of sequences

To reverse a selected block of cells, choose **Reverse** from the **Utilities** menu. For example, if bases 2 through 6 are selected, then base 6 will exchange places with base 2, and base 5 with base 3:



Changing gaps and missing data

The **Change All** submenu of the **Utilities** menu allows you to change cells containing missing data or gaps. The four commands of this sort are (examples to follow):

1. **Terminal Missing to Gaps.** Converts all missing data cells at the start and end of each sequence (before and after any actual nucleotides or amino acids) to gaps. This command does not alter internal missing data cells in the midst of nucleotides or amino acids. It acts on all sequences, whether or not they are selected.
 2. **Terminal Gaps to Missing.** Similar to Terminal Missing to Gaps, but does the reverse, converting all terminal gaps to missing data.
 3. **N to Missing.** For nucleotide data, converts any cells with N to standard missing data.
 4. **Missing to N.** For nucleotide data, converts any cells with missing data to N.

For example, imagine you began with the following sequences:

| | |
|--------------------------|---|
| <i>Pterostichus</i> | -----??TCCTGCACAA |
| <i>Mecyclothorax</i> | -----??TTCCCTGCACAA |
| <i>Amblytelus</i> | -----??TCCTGCACAA |
| <i>Melisodera</i> | -----??CTCTTGACAA |
| <i>Asaphidion curtum</i> | -----TTTGA?TGCAAGTGTATGGTATGTCCTGGCTCCTGCACAA |

Terminal Missing to Gaps would cause all the leading missing data to become gaps:

| | |
|-------------------|--|
| Pterostichus | TCCTGCACAA |
| Mecyclothorax | TTCCCTGCACAA |
| Amblytelus | TCCTGCACAA |
| Melisodera | CTCTTGACAA |
| Asaphidion curtum | TTTGA?TGCAAGTGTCAATGGTATGTCCTGGCTCCTGCACAA |

Terminal Gaps to Missing would convert all the leading gaps to missing data:

| | |
|-------------------|---|
| Pterostichus | ??TCCTGCA CA |
| Mecyclothorax | ??TTCC TGCACA |
| Amblytelus | ??TCCTGCA CA |
| Melisodera | ??CTCTTG CA |
| Asaphidion curtum | ????TTTGAGTGTCATGGTATGTCTGGCTCC TGCACA |

Missing to N would then convert all the ? to N:

| | |
|-------------------|---|
| Pterostichus | NNNTCCTGCA CA |
| Mecyclothorax | NNNT TCCTGCA CA |
| Amblytelus | NNNT TCCTGCA CA |
| Melisodera | NNN CTCTTG CA |
| Asaphidion curtum | NNNNNTTGTGANTGCAAGTGTCATGGTATGTCTGGCTCC TGCACA |

and **N to Missing** would do the reverse.

Removing partial triplets at the start or end of nucleotide sequences

Terminal Part Triplets to Gaps in the **Change All** submenu of the **Utilities** menu converts any partial codons at the start or end of sequences to gaps. In the following example, Bemb. levettei's sequence begins with the partial triplet TT, Bemb.coxendix the partial triplet C, Bemb.harpaloides AA, and Bemb.insulatum T:

| | | |
|----|------------------|-----------------------|
| 10 | Bemb.levettei | - - - - - TTGAGTGCAAG |
| 11 | Bemb.coxendix | - - - - - CGAGTGTAAG |
| 12 | Bemb.harpaloides | - - - - - AATGCAAG |
| 13 | Bemb.insulatum | - - - - - TGAGTGTAAG |

Choosing **Terminal Part Triplets to Gaps** would remove these stray nucleotides:

| | | |
|----|------------------|---------------------|
| 10 | Bemb.levettei | - - - - - GAGTGCAAG |
| 11 | Bemb.coxendix | - - - - - GAGTGTAAG |
| 12 | Bemb.harpaloides | - - - - - TGCAAG |
| 13 | Bemb.insulatum | - - - - - GAGTGTAAG |

This can be convenient before permanent translation of the sequence to protein ("Translating nucleotide sequences to amino acid sequences" on page 307).

Uninformative sites

You may want to have MacClade ignore or delete parsimony-uninformative sites so that they do not get involved in the various calculations performed. "Character exclusion" on page 278 describes the features for including, excluding, and deleting characters. If you want to exclude the uninformative characters, you

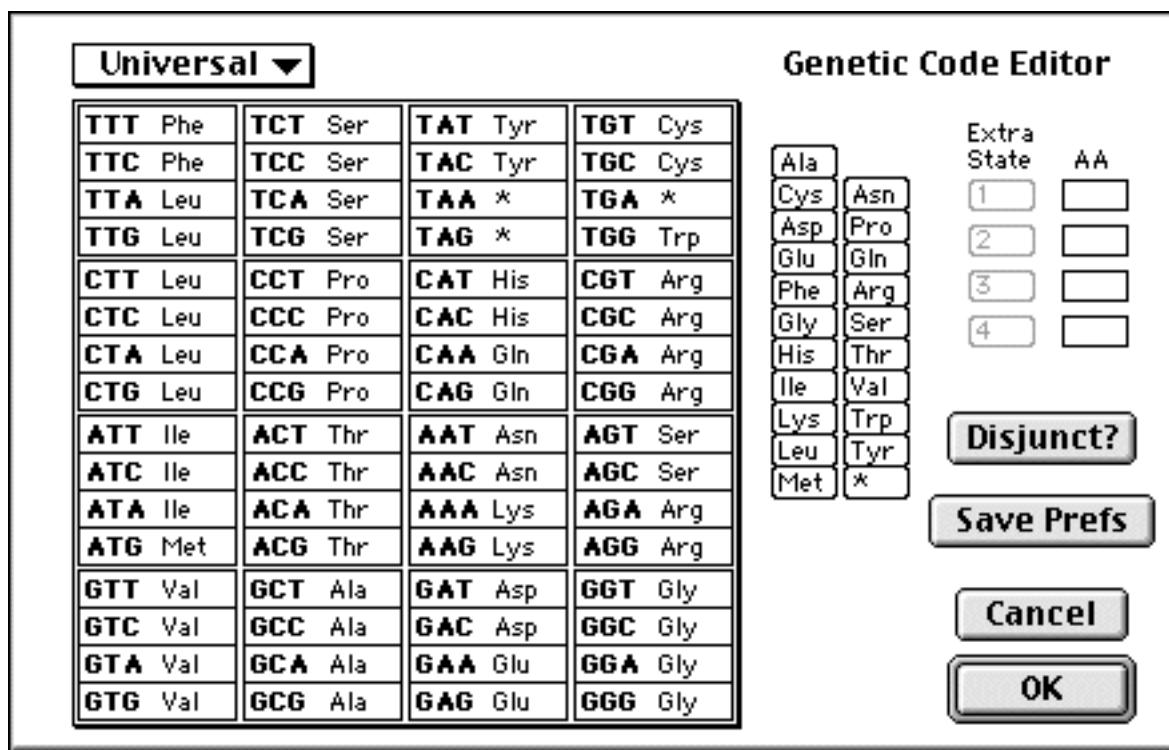
can use **Exclude Uninformative** ([page 279](#)) in the tree window. However, this does not save any memory usage, because MacClade (unlike PAUP*) does not pack the data matrix according to assumptions and exclusions. If you want to get rid of the uninformative characters to save memory, you need to delete them from the data file entirely. To do this, first exclude them using the **Exclude Uninformative** command, and then select the excluded characters in the character list window (see ["Selecting or de-selecting sets of objects" on page 166](#)), and press the Delete key.

Before deleting any characters, you may want to assign names to the characters corresponding to their positions in the sequence, so that their original positions will still be indicated even after other sites are deleted. The **Fill** command will allow you to name the characters automatically in sequence (see "[Naming sites](#)", earlier in this chapter).

Setting the genetic code

MacClade allows the user to specify a genetic code, which MacClade uses for DNA/RNA-to-protein conversion and to create user-defined types of the minimum number of nucleotide changes underlying amino acid substitutions (Swofford and Olsen, 1990; Felsenstein, 1991).

The genetic code can be specified in the **Genetic Code** dialog box from the **Characters** menu:



Several predefined genetic codes are chosen by the pop-up menu at the upper left:

- Universal**
- Universal (extended)**
- Drosophila mtDNA**
- Drosophila mtDNA (extended)**
- Mammal mtDNA**
- Mammal mtDNA (extended)**
- Yeast mtDNA**
- Mold, Protozoan mtDNA**
- Echinoderm mtDNA**
- Flatworm mtDNA**
- Ciliate Nuclear**
- Candida Nuclear**
- (Custom)**

See Elzanowski and Ostell (2000) for descriptions of these and other genetic codes.

The extended codes are those in which the disjunct amino acids are split into distinct states, as described below.

To change the amino acid corresponding to a particular nucleotide triplet, select the nucleotide triplet in the table, then touch on the amino acid name (the little buttons on the right) to which you wish the triplet to correspond. ("*" refers to the termination codon.) Multiple nucleotide triplets can be selected if you hold down the Shift key while clicking on additional triplets.

| | | |
|-----|-------|------|
| TCA | 1.Ser | His |
| TCC | 1.Ser | Ileu |
| TCG | 1.Ser | Lys |
| TCT | 1.Ser | Leu |
| TGA | * | Met |
| | | Asn |

Changing TCG so that it codes for Lys

An amino acid can be said to be disjunct if there exist at least two triplets that code for the amino acid, such that there is no chain of triplets linking them, with adjacent elements in the chain differing by only one single nucleotide mutation, and with all elements of the chain coding for that amino acid. Note that serine is disjunct in the standard nuclear code (as the serine codons AGC and AGT are at least two mutational steps away from serines TCA, TCC, TCG, and TCT). Evolutionary changes between serine AGC and serine TCT therefore must go through an intermediate amino acid. To check to see if the code as you have it currently defined has any disjunct amino acids, touch on the Disjunct? button. Even though serine is disjunct, the Disjunct? button will not indicate this for the standard nuclear code because the two disjunct pieces have been coded as distinct states 1 and 2. (Serines AGC and AGT are coded as 1.Ser, and TCA, TCC, TCG, and TCT are coded as 2.Ser.)

If you create a custom genetic code, there may be disjunct amino acids other than serine. You may wish to code the disjunct pieces of an amino acid as distinct states, as is done automatically for serine with the standard nuclear code. This can be of value if you wish to analyze your data using protein parsimony (see

next section). To do this, you need to first define extra amino acid states. You can define extra states by clicking on a box below the title "AA" beside "Extra State", then clicking on the list of amino acids to indicate what amino acid the extra state represents. Once these extra amino acids are defined, you can then apply them to the genetic code.

If some elements of the code specify extra states (as is the case for ACC, AGT, TCA, TCC, TCG, and TCT in the standard nuclear code), then when you ask MacClade to convert from nucleotide to amino acid sequences, it will use those extra states in the conversion. Thus, if you are using the standard nuclear code, your converted matrix may have some "1" and "2" entries, indicating the two serine states.

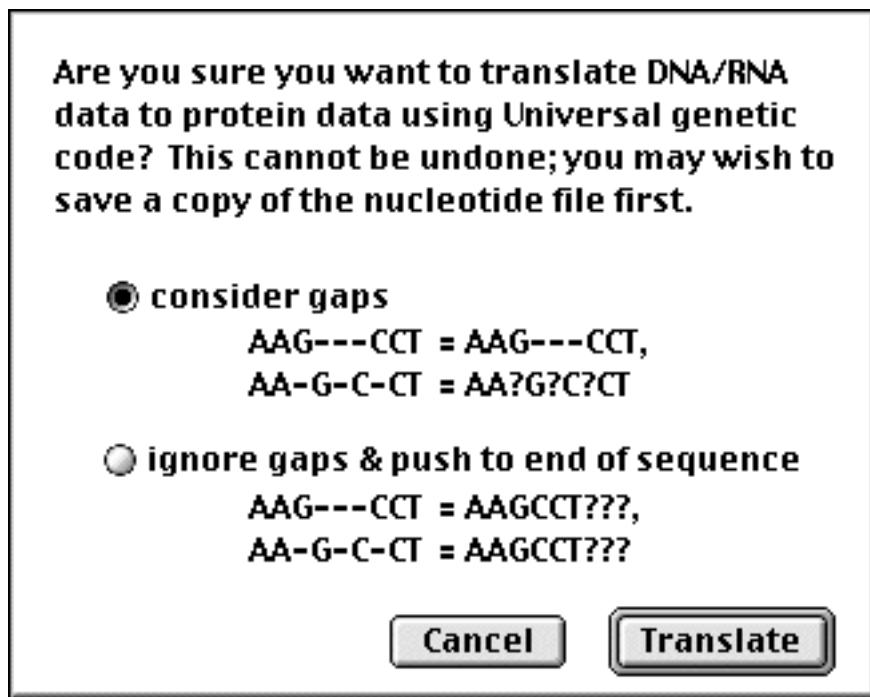
If you modify the genetic code, your new code will be stored in the data file when you save it. There is currently no easy way to make a central storage place for various genetic codes you have created, but if you regularly work with data files that use codes other than MacClade's built-in codes, you may want to make a series of stationery pads (["Saving files to disk" on page 121](#)), each one of which has stored in it a different genetic code appropriate for the different molecules on which you work.

WARNING: *If you wish to export your data matrix to a program that doesn't understand that the symbols 1 and 2 mean serine, then use **Recode** in the **Utilities** menu to convert 1 and 2 to S before you export your file.*

Translating nucleotide sequences to amino acid sequences

If your data are DNA or RNA sequences, and you select the **Translate to Protein** item in the **Data Format** submenu of the **Characters** menu in the data editor, then MacClade will transcribe and translate the nucleotide sequence into an amino acid sequence according to the current genetic code. (If in doubt about the nature of the genetic code, and therefore of the translation used by MacClade, open the **Genetic Code** dialog box from the **Characters** menu, which specifies the correspondence between nucleotide triplets as they appear in the matrix and amino acids.)

When you ask MacClade to translate nucleotide data to amino acid data, you will be presented with the **Translate to Protein** dialog box:



If gaps are considered, then three consecutive gaps in positions 1, 2, and 3 are taken to mean that there is a gap in the amino acid sequence; other gaps are considered equivalent to missing data. In translating a triplet containing missing data, MacClade will convert to all possible amino acids consistent with the partially specified triplet. For example, "CA?" will be treated as equivalent to "CAA or CAC or CAG or CAT", and will be converted to Gln/His. If gaps are ignored, then MacClade will skip over gaps during translation. Considering gaps essentially preserves some of the information in the alignment of nucleotides; ignoring gaps essentially removes the alignment.

MacClade's translation is the same whether the data file is RNA or DNA; it does not invert the sense of the codons for one or the other. When in doubt, check the **Genetic Code** dialog box to see how the translation will be performed.

MacClade will automatically name the resulting protein sites. Sites translated from the first protein-coding block in the file will be named 1.1, 1.2, 1.3, and so on. If in the original nucleotide sequence there was more than one protein-coding region (separated by non-coding regions), then sites in the second translated region would be named 2.2, 2.3, 2.4, and so on.

Some sequences cannot be translated by MacClade. For example, if three consecutive sites are designated as positions 1, 2, and 2, then MacClade will recognize that adjacent sites do not have adjacent codon positions and will refuse to translate. MacClade will also refuse to translate if a non-coding site is followed by a site designated as position 2 or 3.

In translating the data, all type sets, weight sets, and inclusion sets are destroyed, as are all footnotes and pictures (except those attached to taxon names), and all character names. All characters are reset to the default type.

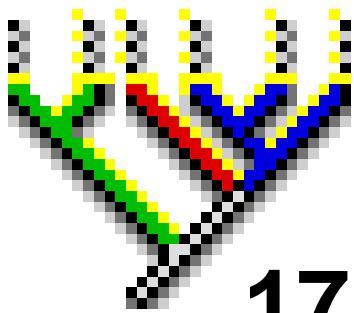
Creating protein-parsimony user-defined types

To create a user-defined type in which the cost of a change between two amino acids is the minimum number of amino acid changes required to convert between the two amino acids, press the New button in the **Type Edit** dialog box in the **Characters** menu. This can only be done if the data editor is visible. Check the box "Protpars type based on genetic code". If you then press Create, MacClade will use the current genetic code to calculate the minimum number of changes between amino acids, and will create a user-defined type based on these changes. Such a type can be used to implement protein parsimony (Felsenstein, 1991).

WARNING: Your data matrix may contain amino acids that are no longer defined in the current genetic code, perhaps to avoid disjunction. For example, in the standard nuclear code, serine (symbolized by "S") is replaced by serine.1 ("1") and serine.2 ("2"), as there are two disjunct sets of triplets coding for serine (see previous section). If MacClade detects any serine ("S") in your data matrix, or other similar amino acids, as it creates a protein parsimony type, it will ask if you wish to convert these to their non-disjunct equivalents. For serine, this will involve converting all S to 1/2. In the process, any cells containing such amino acids that were polymorphic will be converted to uncertainty (e.g., V&S will be converted to V/1/2).

File saving options

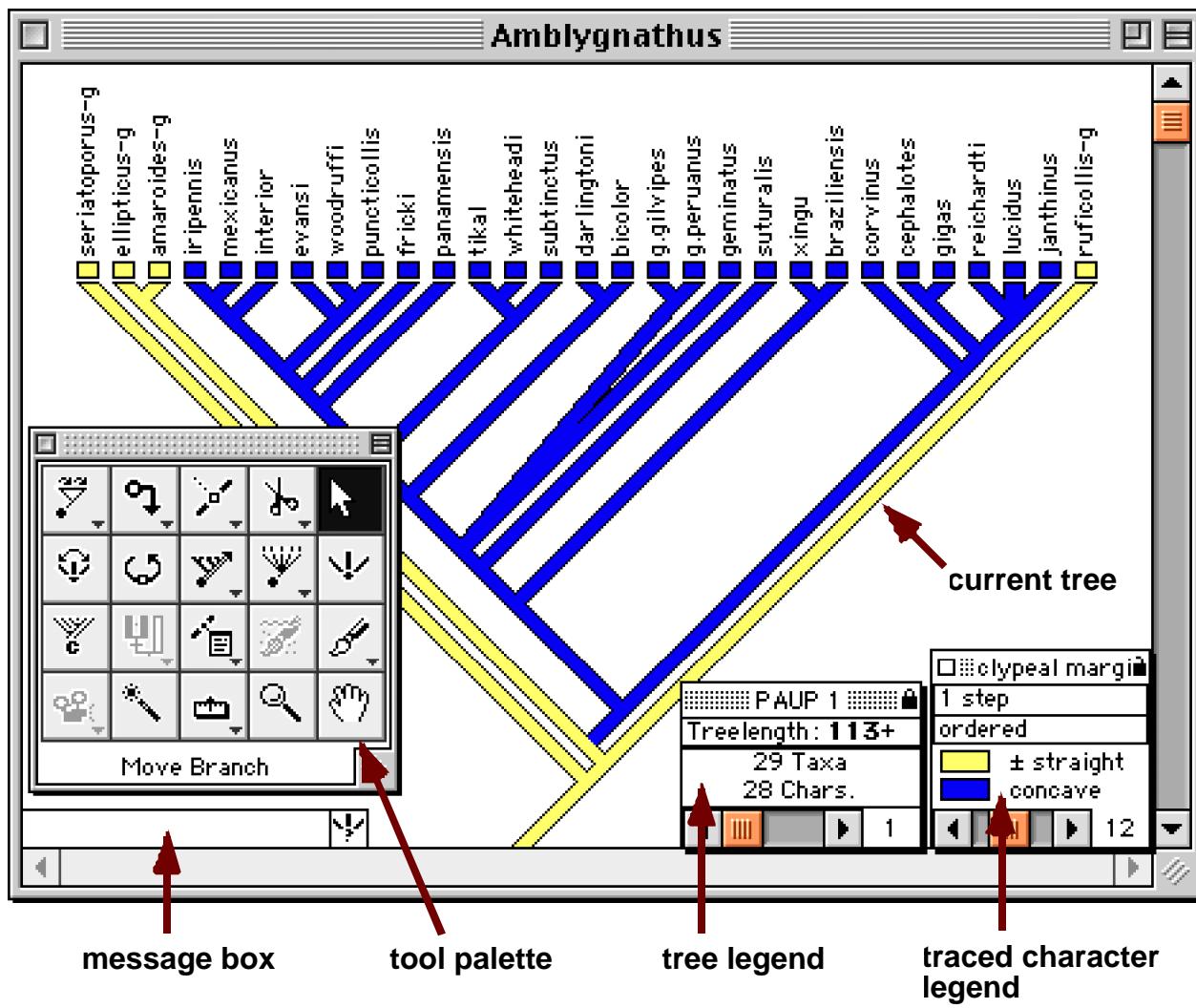
You have some flexibility regarding how MacClade writes NEXUS files. Some of the available features are specifically designed for molecular sequence data. See [Chapter 7](#) for details.



TREES AND TREE MANIPULATION

17

It is in MacClade's tree window that the relationship between hypothesized phylogenetic trees and data is explored. This chapter concerns the basic functions of the tree window, including those for manipulating trees.



Trees in MacClade

Trees in MacClade can be dichotomous, polytomous, or have observed taxa placed as ancestors. They may include all or only some of the taxa in the data matrix. See [Chapter 3](#) for a discussion of trees and their terms.

MacClade can show only one tree at a time in its tree window. However, MacClade can hold multiple trees in its memory and show them one at a time, or cycle through them to perform calculations on each one.

MacClade's tree window

MacClade's tree window displays a tree that can be manipulated and analyzed. The tree window is scrollable, resizable, and has several smaller windows that are associated with it: the tree legend, the trace legend (e.g., if a character is traced), and the tool palette.

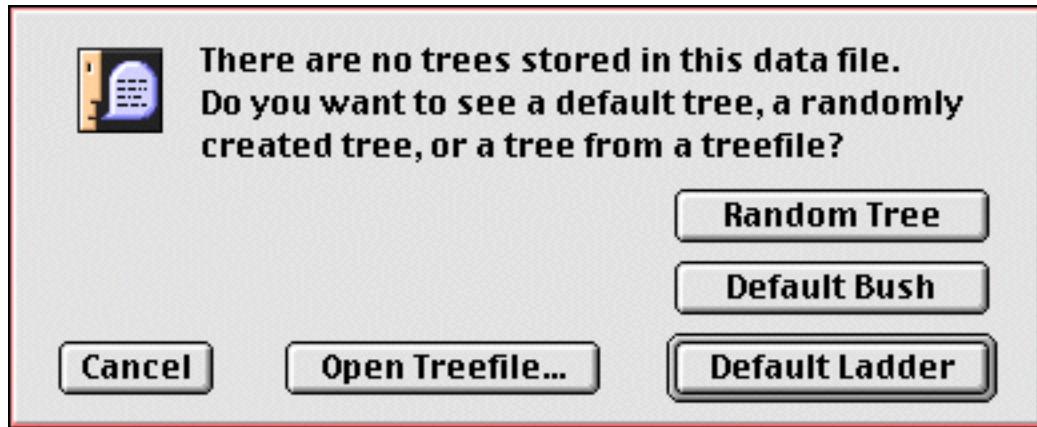
Getting to the tree window

You can move from the data editor to the tree window by choosing **Tree Window** from the **Windows** menu. If the tree window is hidden by another window, **Tree Window** will bring the tree window to the front.

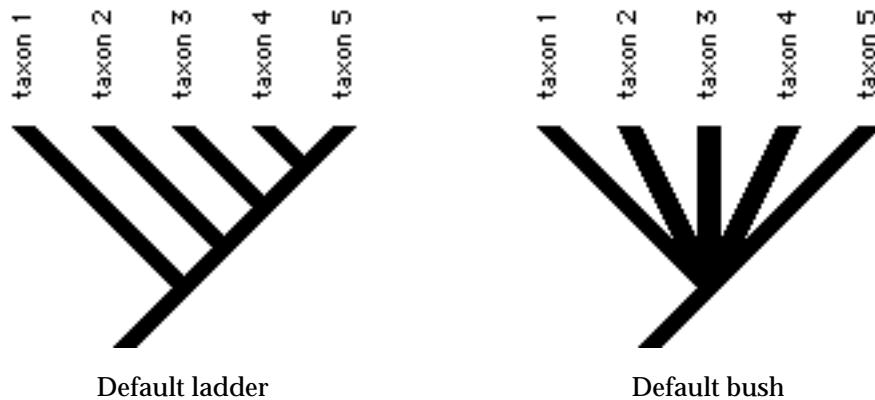
NOTE: *If you hold down the Command key (⌘) as the file is being read in, you will be taken automatically to the tree window, even if the file was last saved with the data editor on the screen.*

The tree window and data editor cannot be open simultaneously. For this reason MacClade 4 is either in tree window mode, or data editor mode. Each of these modes has its own separate menu structure.

In moving to the tree window, MacClade will either take you to a previously saved tree, or ask you to choose a tree. If no trees have been stored in the data file, MacClade will present you with the following dialog box,



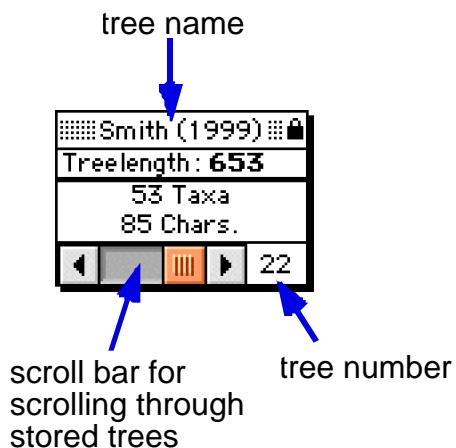
and ask that you choose between the two default trees (a ladder or a bush), or a randomly generated tree, or trees in an external tree file. The ladder is simply a comb in which taxa are arranged according to their position in the data matrix; the bush is a full polytomy with no cladistic structure:



Tree window structure and size

Legends

With a tree on the screen, you will see a small "tree legend" that gives information about the tree on the screen, and allows you to scroll to other stored trees:

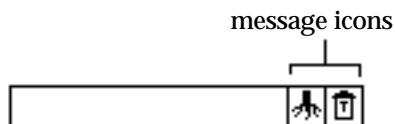


If a character is traced, a legend window will appear, as described in [Chapter 18](#) for discrete characters, or [Chapter 21](#) for continuous characters. If Trace All Changes is shown, it will also have a legend.

By default, these various legends are locked; that is, they hug the lower-right corner of the tree window. If you unlock them (turn off **Lock Legends** in the **Display** menu), you can then move them around independently of the tree window. If the legends are unlocked, they will automatically shrink and expand to accommodate the full length and width of character and state names and the treelength.

Message box

The message box on the lower left of the tree window gives information about many things. If the cursor is over top of a branch, it gives the name of the taxon or says how many taxa are in the clade above the branch.



The message box

On the message box's right-hand side, icons may appear that indicate whether the DELTRAN or ACCTRAN options are in effect, if a branch has a fixed state, and so forth. The icons and their meanings are given in the table below:

| Icon | Meaning | Action if Double-Click | Action if Option-Double-Click |
|------|---|--|-------------------------------------|
| | Some characters excluded | | Includes all characters |
| | Some taxa excluded | Presents Inc-Exc Taxa dialog | Includes all taxa |
| | Clade expanded, thus only part of the tree is visible | | Shows all clades |
| | At least one character of a directed type | | |
| | At least one branch with fixed state in traced character | Shows branches with states fixed | Unfixes all branches |
| | DELTRAN option active in traced character | Presents Resolving Options dialog | Turns off DELTRAN |
| | ACCTRAN option active in traced character | Presents Resolving Options dialog | Turns off ACCTRAN |
| | Polytomies in tree, assumed "soft" (uncertain resolution) | Presents Polytomy Options dialog | Arbitrarily resolves all polytomies |
| | Polytomies in tree, assumed "hard" (multiple speciation) | Presents Polytomy Options dialog | Arbitrarily resolves all polytomies |

The icons placed beside the message box can be used, in some circumstances, as buttons to perform actions. By double-clicking on the icon, the actions shown in the third column of the table above are performed. Actions in the fourth column are performed when the Option key is held down while double-clicking.

NOTE: To see amount of unused memory within MacClade's memory, click on the message box.

Tree window size

When MacClade first begins on an old, small-screened Macintosh (Macintosh Plus and the equivalent), the tree window fills the screen, and the title bar and scroll boxes are not visible. To expose the title bar at the top and grow box at the lower right choose the **Reduce Window** menu item under **Display**. Once reduced, the window is moveable and resizable. On all modern Macintoshes, however, the tree window will be automatically positioned so that the title bar and grow box will be visible.

If the window is reduced so that the title bar and grow box are visible, choosing **Expand Window** from the **Display** menu will expand the window so that it fills the screen.

If you prefer an expanded tree window on a larger screen or reduced windows on a small screen, you can set those preferences using the **Save Preferences** dialog from the **File** menu.

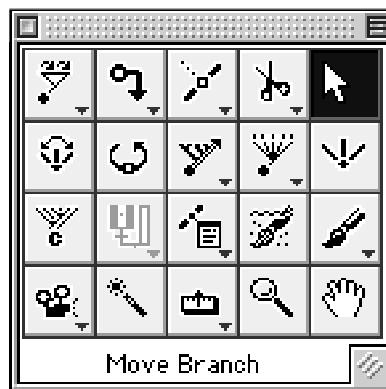
Moving over the tree

If the tree is too big to fit into the window, you can move the view around using the hand tool (from the tool palette. If the tree window is in "reduced" mode (i.e., you can see the title bar and scroll bars), then you can scroll through the tree using the scroll bars. If you grab a branch and try to drop it on a branch off the screen, the tree will automatically scroll. If the distance you want to move is too far for the window to scroll automatically, you can use the branch storage feature ([page 329](#)).

NOTE: Whatever the current tool, you can temporarily turn it into the hand tool by holding down the Shift key.

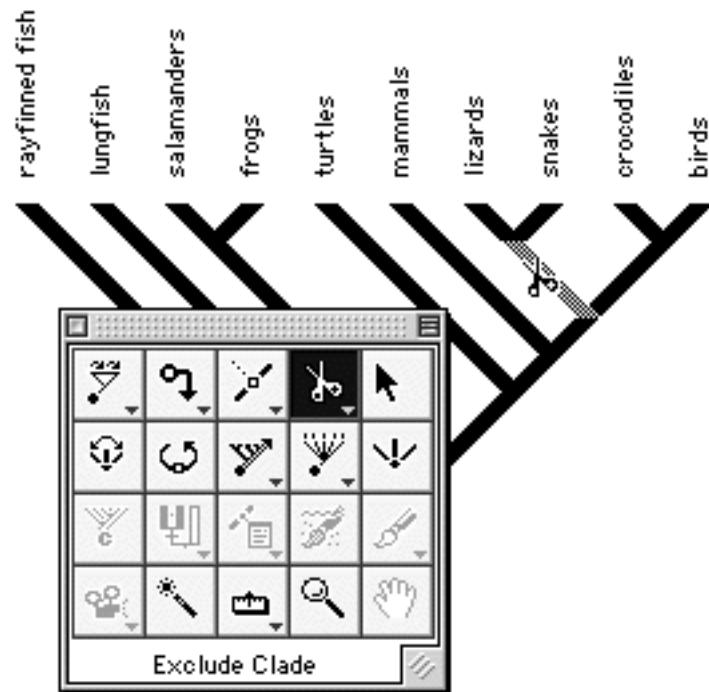
The Tool Palette

The tree window's tool palette (see [page 158](#)), shown below, provides a set of tools for manipulating tree structure and for examining character evolution.

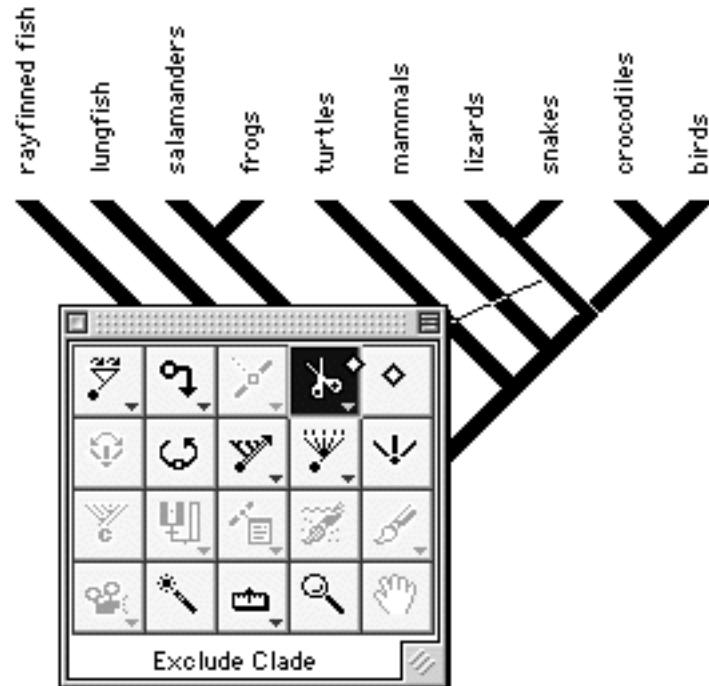


You can use it in two ways. The standard method, shown below, is to use the palette is exactly like the palette in many other programs: Move the cursor over the palette and select the icon of your choice. The cur-

sor will then become the tool pictured by that icon, and you can move the tool over a branch and click. The command corresponding to that tool will be performed on the branch.



The other method is to use the standard arrow tool to grab a branch and drop it onto one of the icons of the palette. As you move the diamond cursor over each icon, the command that would be performed by that icon is listed in the small message box below the palette. If a command is illegal for a given branch, its icon may be dimmed.



NOTE: As you hold the current tool over a branch, the message bar at the lower left of the tree window indicates what branch you are over — either by the name of the terminal taxon, or by the number of taxa in the clade above the branch. This message helps you know if the tool is over the branch intended.

NOTE: Whatever the current tool, you can temporarily turn it into the arrow tool by holding down the Command key (⌘).

The tools available in the palette are briefly described in the table below.

| Tool | Description | Tool | Description |
|------|---|------|---|
| | Move Branch tool. The standard tool for moving branches. See " Moving branches and clades around " on page 317. | | Fix States tool. For the traced character, allows you to fix the state of a branch. See " Fixing the state of a branch " on page 350. |
| | Exclude Clade tool. The scissors allow you to remove taxa above or below a branch from the tree. See " Including and excluding taxa " on page 327. | | Remove Fix tool. Unfixes the state of a branch for the traced character, allowing MacClade to assign the most parsimonious state to the branch. See " Unfixing state of a branch " on page 353. |
| | Make Ancestral tool. Designates taxa as ancestors or terminals; in search mode searches for optimal designations. See " Searching for optimal ancestors " on page 332. | | Branch List tool. Lists information about a branch or a change along a branch when Trace All Changes or Trace All States is shown. See " More information about the changes occurring on branch " on page 367 and " Summary of all states along each branch " on page 371. |
| | Reroot Clade tool. Reroots the entire tree or just a clade. See " Reroooting clades " on page 319. | | Temporary Trace tool. Temporarily traces a character displayed in Trace All Changes. See " Tracing a character shown in Trace All Changes " on page 368. |
| | Search tool. Rearranges the branches of the tree in an attempt to find a more parsimonious tree. See " Searching for short trees " on page 330. | | Test Correlation tool. Implements the Concentrated Changes test. See " The concentrated-changes test " on page 423. |
| | Collapse Branch tool. Removes a branch from the tree, thereby creating a polytomy. See " Polytomies " on page 324. | | Move Tree tool. For trees that take up more room than is available within the tree window, allows you to move through the tree. See " Moving over the tree " on page 314. |

| Tool | Description | Tool | Description |
|------|---|------|---|
| | Collapse Clade tool. Removes all resolution above or below a branch. See " Polytomies" on page 324. | | Expand Clade tool. Magnifies a portion of the tree to fill the tree window. See " "Expanding and shrinking" on page 336. |
| | Ladderize tool. Rotates nodes to maximize rightward or leftward lean of the tree. See " "Ladderizing clades" on page 335. | | Set Evolve Segments tool. Sets the number of segments on a branch for use with the Evolve Characters simulator. See " "Setting branch lengths: Evolve segments" on page 437. |
| | Rotate Branches tool. Rotates nodes on a tree. See " "Rotating branches" on page 334. | | Selection Wand. Selects in a list window the taxa present in a clade, or the characters that change along a branch. See " "Selecting all taxa in a clade" on page 173 and " "Selecting all characters that change along a branch" on page 173. |
| | Polytomy Exchange tool. Exchanges places of elements within a polytomy. See " "Exchanging elements of a polytomy" on page 335. | | Show Picture tool. Allows you to see, in the tree window, pictures attached to taxon names or matrix cells. See " "Show Picture tool" on page 446. |

In the following sections, the action of each tool is documented.

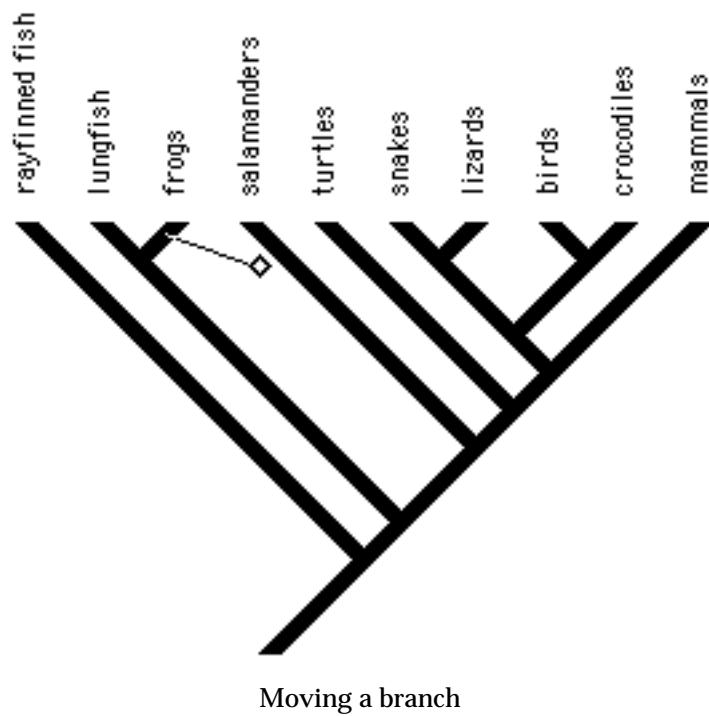
Tree manipulation: Changing cladistic structure

The following commands substantively change the tree. You can Undo most of these commands; see "Undoing tree manipulations" at the end of this section.

Moving branches and clades around

With the arrow tool, a branch (and along with it any clade above it) can be picked up and moved to another part of the tree.

Select a branch to be moved by clicking on it with the arrow tool and holding the mouse button down; the cursor will change from an arrow to a small diamond. Drag the diamond to the place you want the branch dropped, and drop it by releasing the mouse button.

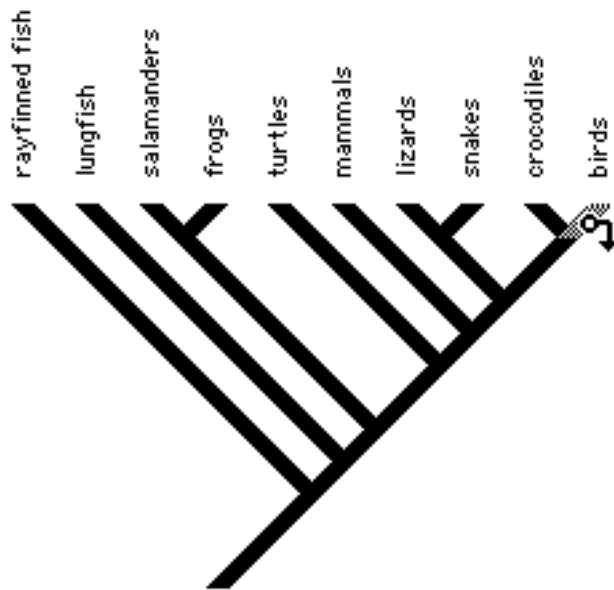


If the tree is very large and extends over several screen widths, you may want to use the branch storage feature for a long distance branch move (see [page 329](#)).

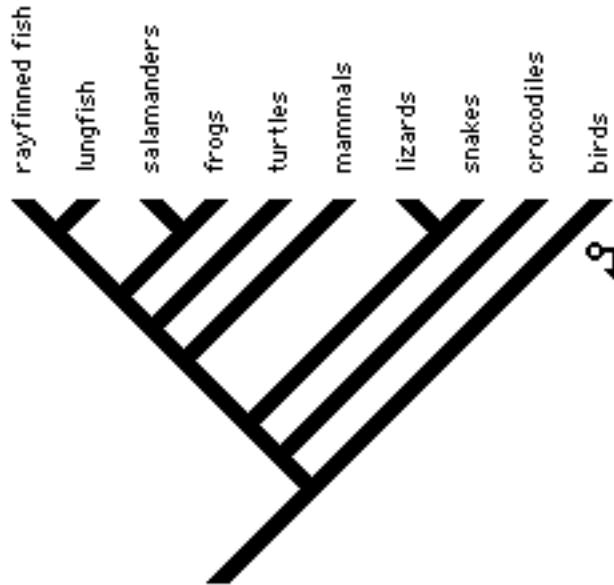
In some circumstances MacClade will refuse to move a branch because it would be inappropriate (for instance, if you try to move a branch onto one of its descendants). After a branch move, MacClade will automatically recalculate treelength, consistency index, and so on, for the new tree. If any branches involved in the move have their states fixed in the traced character, their states will be unfixed by the move and a warning given.

Rerooting clades

Choose the Reroot tool and click on the branch where the tree or clade is to be rerooted. For example, clicking on the branch leading to birds



will cause the current root of the tree to be removed, the tree pulled down by the branch between birds and the other taxa, and a new root placed there.



Rerooting may unfix fixed states at nodes and will make any observed ancestral taxa terminal again (MacClade will give warnings if these occur).

Rerooting the entire tree will not change the treelength unless some characters are irreversible, strati-

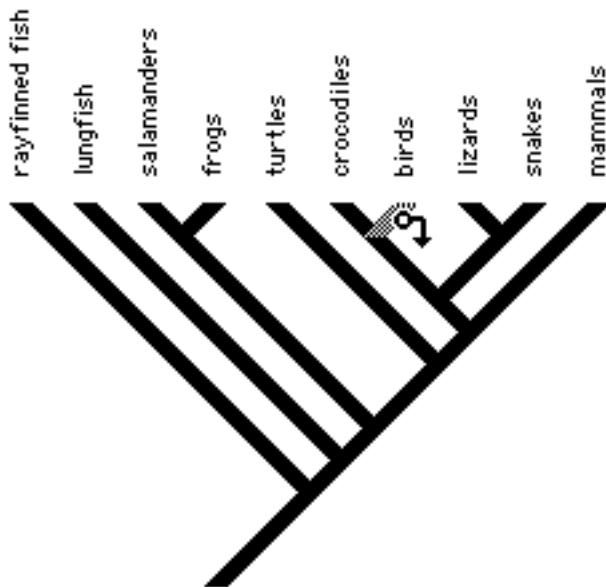
graphic, or of a user-defined type with an asymmetrical step matrix. Rerooting can, of course, have an impact on the tracing of character evolution and other derived statistics.



EXAMPLE: Try rerooting the tree presented in the example file "Rooting Example"; note that the treelength is the same under all rootings. Now select the type set "All irreversible" from the **Type Set List** window (in the **Type Sets** submenu of the **Characters** menu, and press the **Use** button to invoke the type set "All irreversible". Now try rerooting the tree. Note that the length of the tree varies between rootings.

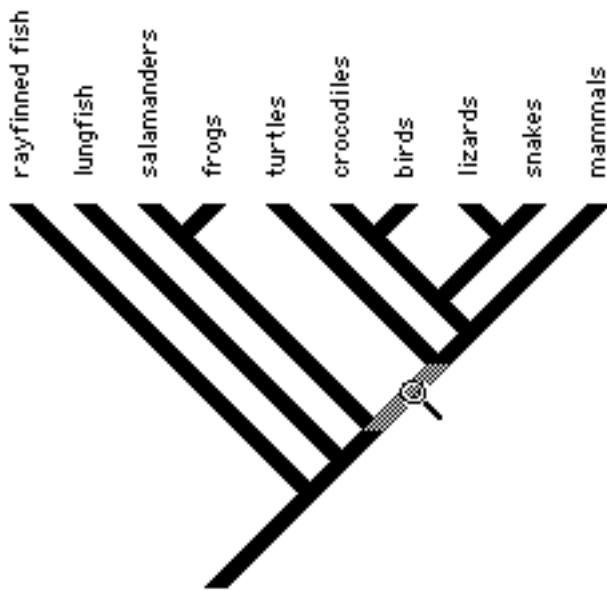
Rerooting within clades

If you ask MacClade to reroot the following tree by clicking on birds, as shown below, then the entire tree will be rerooted at birds.

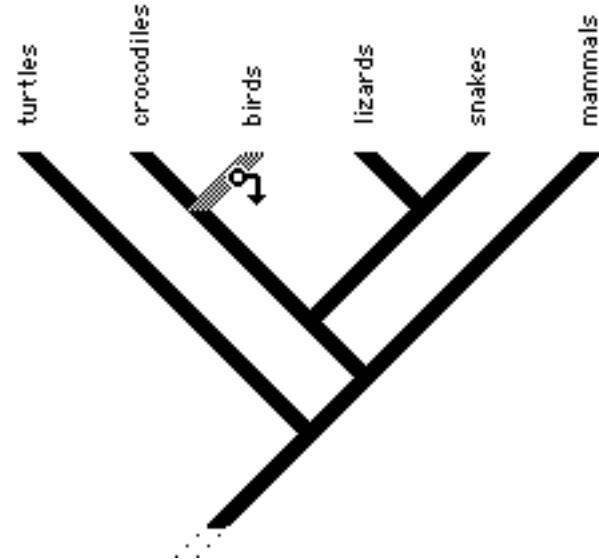


But what if you wish to reroot not the whole tree, but instead only the amniotes? This can be useful, for example, if you wish to try different rootings of the ingroup. MacClade provides two methods that can be used to reroot within individual clades (as opposed to rerooting the whole tree).

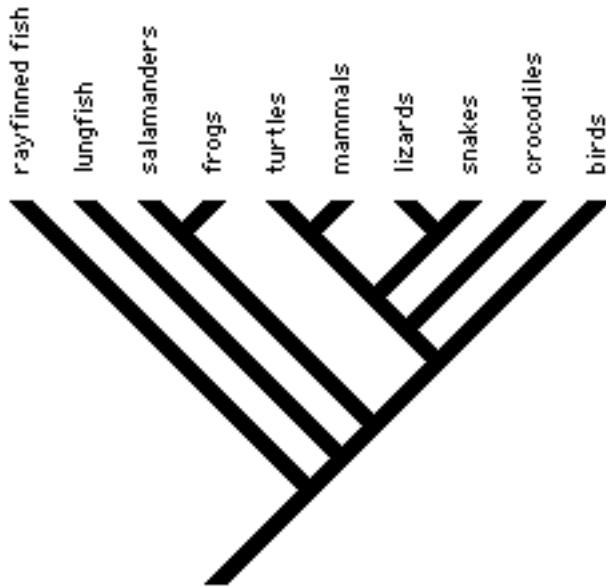
First, you can expand the clade of interest to fill the tree window, by using the expand tool:



Once the clade has been expanded to fill the tree window (as indicated by the small magnifying glass near the lower-left corner of the tree window), then the rerooting tool reroots *only* the clade shown in the tree window:

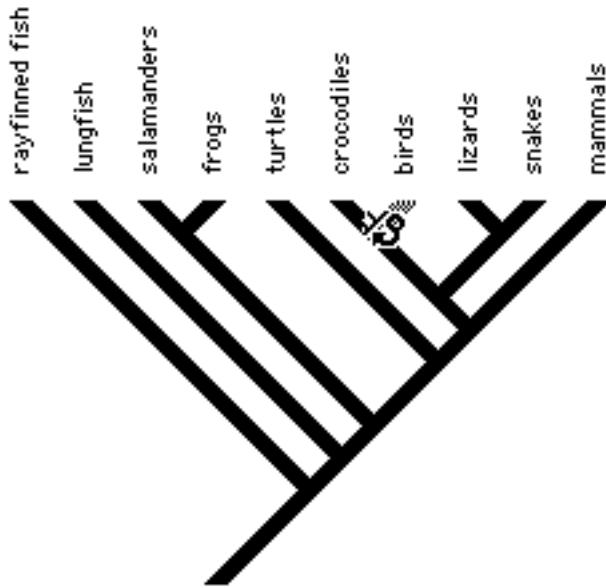


The rest of the tree below the clade remains untouched. Examining the entire tree, we can see that only the amniotes were rerooted:

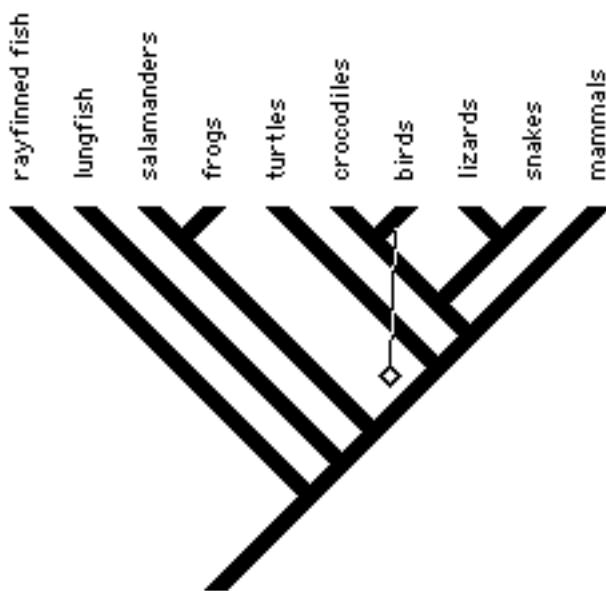


The second approach is to change the mode of action of the reroot tool. By default, the reroot tool reroots the entire visible tree. But the reroot tool's pop-up menu allows you to change the mode of action so that it reroots only within a clade. (You can also switch modes temporarily by holding down the Option key.)

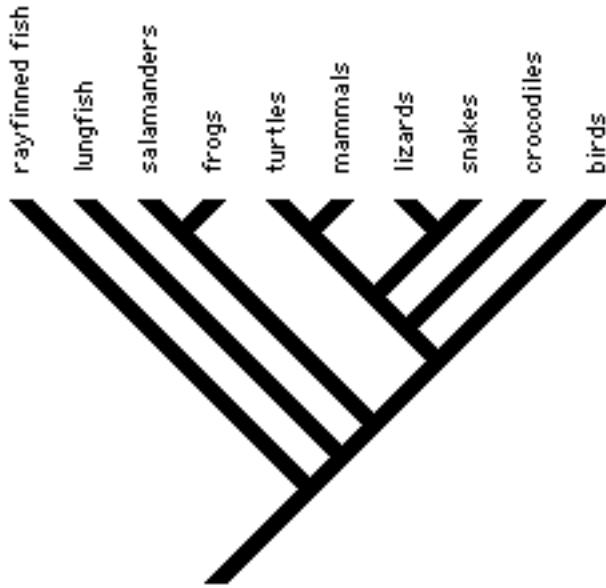
In this mode, if you click on a branch, such as birds



and drag the branch onto the root of a clade that contains birds,



then that subclade will be rerooted, so that the new root intersects the bird branch:



All possible rootings

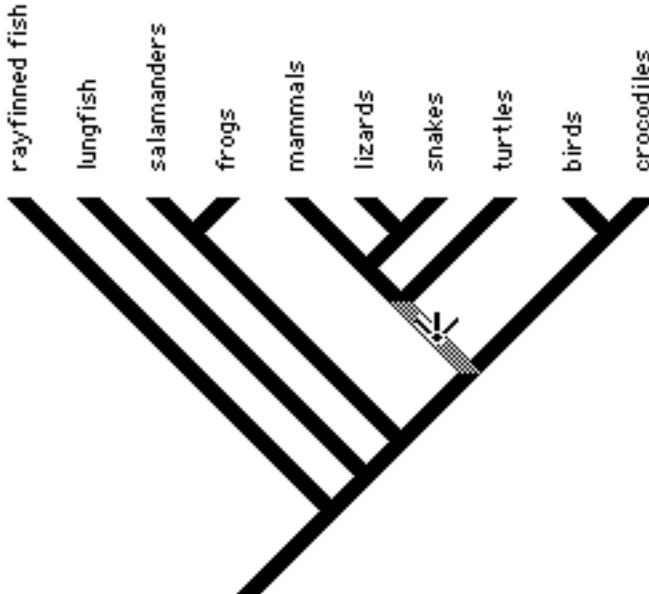
If you wish to save all possible rootings of a clade, then expand the clade to fill the tree window, and choose **All Rootings** from the **Create Trees** submenu of the **Trees** menu. MacClade will then store in the current data file (or tree file, if one is active) trees representing all possible rootings of the clade. To make this collection of trees, MacClade reroots the clade in turn at each of the branches of the clade. Each of the resulting trees is given a name whose suffix is the number of the branch at which the clade was rerooted. This allows you, for instance, to make a chart of the lengths of the rerooted trees, selecting the "length of each tree to log file" option, and then relating the text file output to the branch numbers (available in the

Display menu) to see the cost of various rootings.

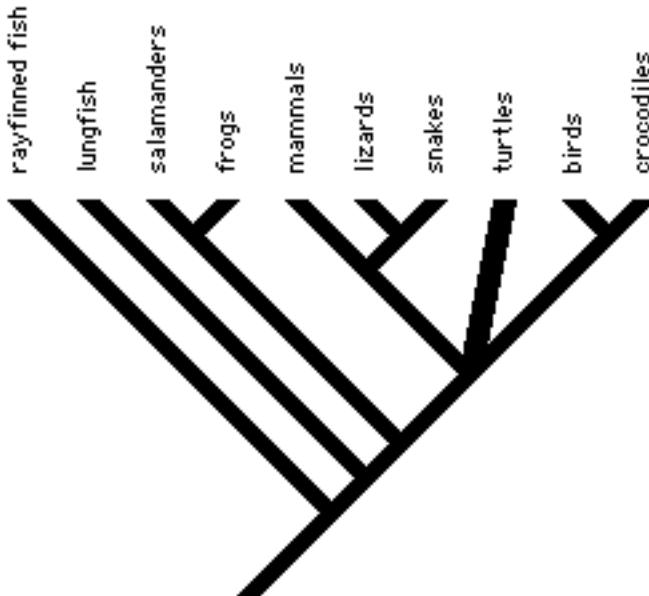
Polytomies

Polytomies can be produced using the collapse-clade () and collapse-branch () tools. Collapse clade makes the whole clade above the selected branch an unresolved bush; collapse branch destroys the resolution indicated by the selected branch, and thus yields a polytomy. These are undoable.

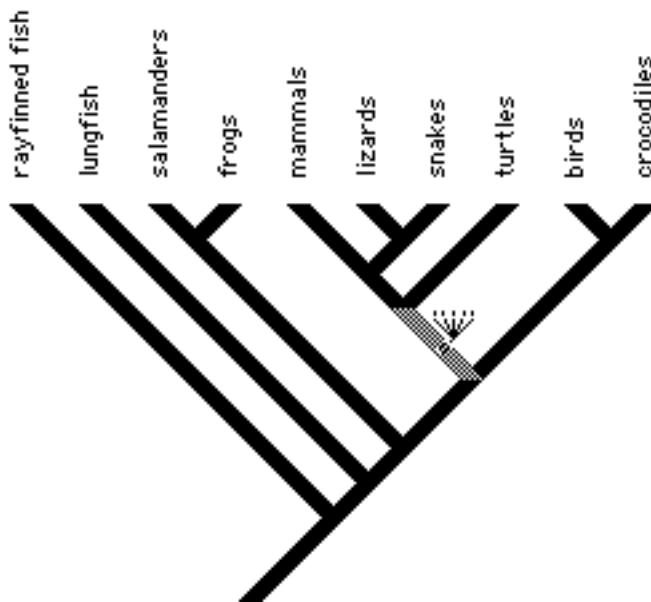
For example, clicking the collapse-branch tool as shown below



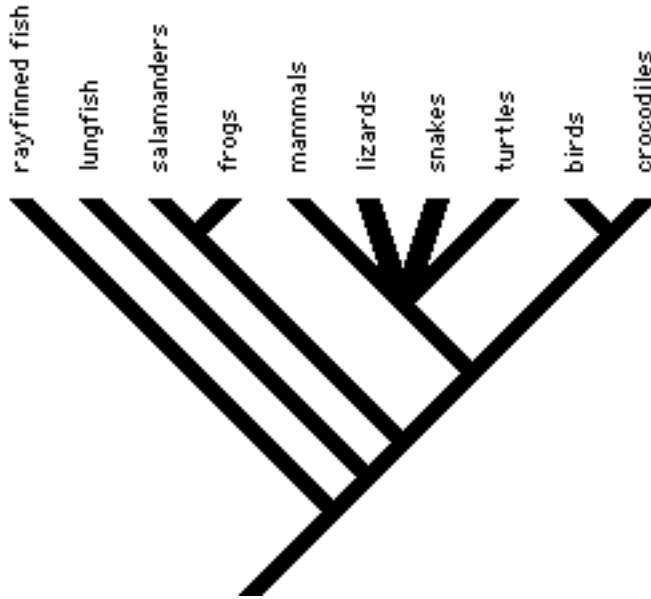
would destroy the branch, and yield the trichotomy shown below:



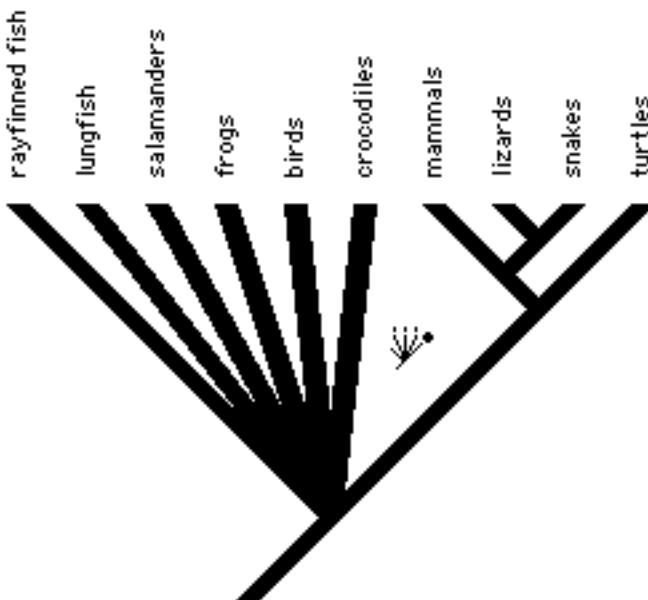
Clicking the collapse-clade tool on the same branch



would destroy all resolution in the clade *above* the branch:



By default, the collapse-clade tool collapses the entire clade above the branch; if you wish it to collapse the entire clade below the branch, then either switch the tool's mode using its pop-up menu, or hold down the Option key while you use the tool. In this mode, the whole tree **below** the branch will collapse into an unresolved bush; the clade above the branch will be unaffected, as shown in the following figure.



If you have expanded a portion of the tree to fill the tree window (as indicated by the small magnifying glass (🔍) near the lower-left corner of the tree window), so that you are looking at just one clade, then the collapse-clade tool in collapse-below mode will collapse those branches *in the visible clade* below the branch.

Soft and hard polytomies

MacClade offers two ways to interpret polytomies: (1) as regions of ambiguous resolution ("soft" polytomies), or (2) as multiple speciation events ("hard" polytomies). The former is probably the more usual interpretation for phylogenetic trees. These two interpretations are discussed more in [Chapter 4](#) and by W. Maddison (1989).

Before you fill your trees with an abundance of polytomies, you should understand clearly what they mean and their problems. (See W. Maddison, 1989, and [Chapter 4](#) herein; note that the interpretation of treelength is especially problematic.) Soft and hard polytomies differ in the interpretation of their display, as discussed in "Interpreting the character tracing", in [Chapter 18](#). You may often find it best to examine various dichotomous resolutions of the polytomy, either by hand or generating them randomly using MacClade's random resolutions feature ([Chapter 23](#)).

MacClade's default interpretation is soft (uncertainty) because that is typically what biologists intend their polytomies to mean. The interpretation used by MacClade can be changed in the **Polytomy Options** dialog box in the **Trees** menu. Whether polytomies are soft or hard is indicated by an icon in the message box

on the lower left of the tree window. The icon will appear if soft polytomies are present, the icon if hard.

Restrictions on polytomous trees

MacClade prohibits polytomies when some characters are of a user-defined, irreversible, stratigraphic, or Dollo type, or when some taxa are fixed as ancestors, because of computational difficulties. Note that PAUP* does not make these prohibitions: user-defined, irreversible, and Dollo types are allowed with polytomies. The reason for this is that PAUP* has only the "hard" variety of polytomies (multiple-specia-

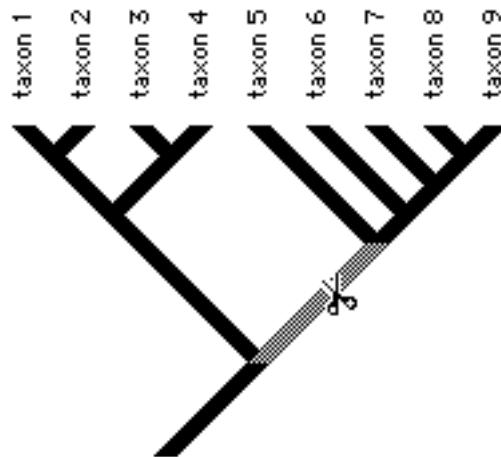
tion), for which algorithms are relatively easy to write. It has proven more difficult to write algorithms for these character types for "soft" polytomies, which are the default for MacClade. For instance, with a user-defined type character, soft polytomies would require a potentially difficult search for the best resolution of the polytomy for the character. To date, only the unordered and ordered characters have soft polytomy algorithms written.

MacClade also prohibits polytomies when continuous characters are traced, with one exception: It allows hard polytomies when squared-changed parsimony is used, as long as there is no basal polytomy with unrooted squared-changed parsimony.

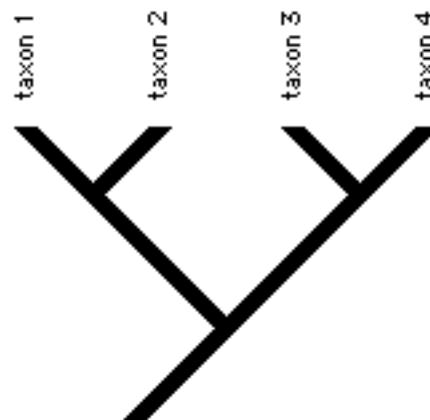
Polytomies prohibit explicit or implicit examination of all MPRs and its uses (namely, Trace All Changes in maximum or minimum-average-maximum mode; State Changes & Stasis chart with mean or minimum and maximum number of changes; Character Steps/etc. chart with numbers of changes shown; **Tree Changes** from the Σ menu).

Including and excluding taxa

Trees need not include all taxa that are present in the data matrix. You can exclude taxa or clades by using the scissors tool from the palette. Clicking the scissors on the branch below the clade will excise the whole clade from the tree. For example, if the 6 taxa on the right are excised

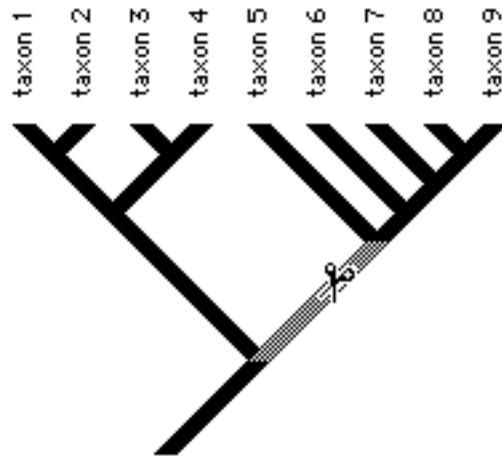


only the four taxa on the left will remain:

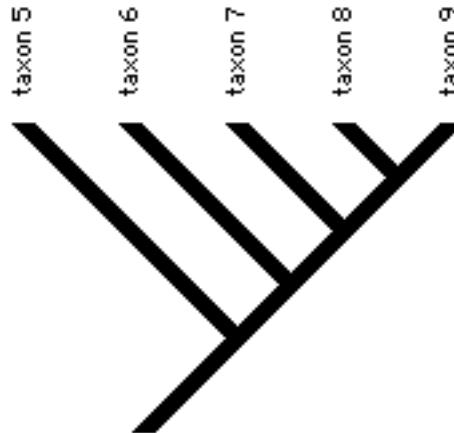


Excluding taxa from trees may cause problems with other computer programs that expect trees will include all taxa in the data matrix. However, it is often useful to keep extra taxa in the data matrix (e.g., extra outgroups) that are only occasionally incorporated into trees.

As with several of the other tools in the tree window, the scissors have another mode, in which all the taxa outside of the clade are removed from the tree. You can switch to this mode using the tool's pop-up menu, or by holding down the Option key. For example, touching the scissors in Exclude-below mode on a branch

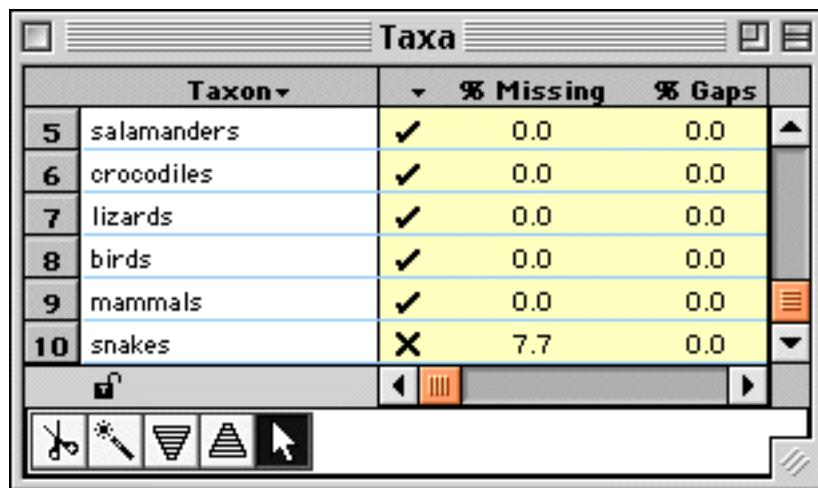


will cause the exclusion of all taxa outside that clade:



If a single clade on the tree is expanded to fill the window (as indicated by the small magnifying glass near the lower-left corner of the tree window), then the scissors in Exclude-below mode will remove all taxa **below** the branch but within the expanded clade.

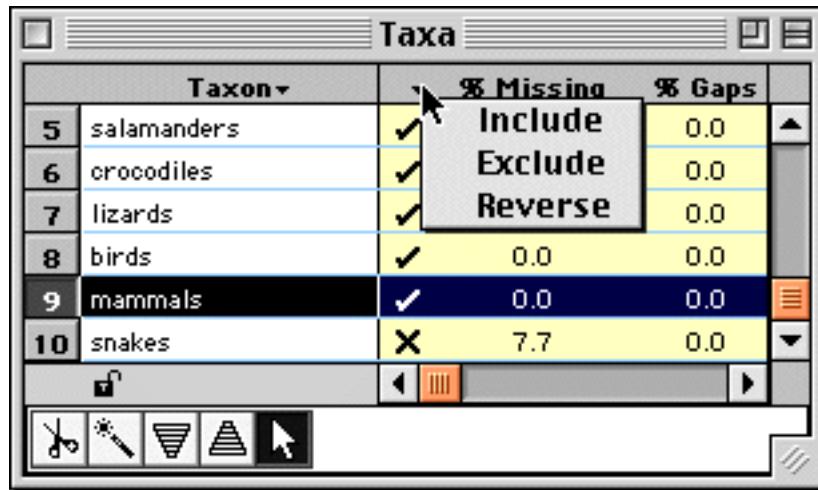
You can also include or exclude taxa from the tree using the taxon list window. Taxa that are excluded from the current tree are indicated by an X in the first column to the right of the taxon name in the taxon list window. For example, the taxon "snakes" is excluded from the tree in the example below:



The screenshot shows the 'Taxa' window in MacClade. The window title is 'Taxa'. It contains a table with three columns: 'Taxon', '% Missing', and '% Gaps'. The rows represent different taxa, each with a taxon number (5, 6, 7, 8, 9, 10) and a name (salamanders, crocodiles, lizards, birds, mammals, snakes). The '% Missing' column has checkmarks for all taxa except 'snakes', which has an 'X'. The '% Gaps' column shows values of 0.0 for all taxa except 'snakes', which is 7.7. The bottom of the window features a toolbar with various icons.

| Taxon | % Missing | % Gaps |
|---------------|-----------|--------|
| 5 salamanders | ✓ | 0.0 |
| 6 crocodiles | ✓ | 0.0 |
| 7 lizards | ✓ | 0.0 |
| 8 birds | ✓ | 0.0 |
| 9 mammals | ✓ | 0.0 |
| 10 snakes | X | 7.7 |

Taxa that are included in the tree are indicated by a check mark. If you wish to exclude taxa, then select them in the taxon list window, and choose **Exclude** from the **Include-Exclude Taxa** submenu in the **Trees** menu, or choose **Exclude** from the menu that pops up when you touch on the little triangle at the top of the column indicating whether taxa are included or excluded in the taxon list window. The latter method is shown below:



This screenshot is similar to the one above, showing the 'Taxa' window in MacClade. However, the context menu is open over the first row ('salamanders'). The menu options are 'Include', 'Exclude', and 'Reverse'. The 'Include' option is highlighted with a gray background, while 'Exclude' and 'Reverse' are in white text on a dark background. The rest of the window and toolbar are identical to the first screenshot.

| Taxon | % Missing | % Gaps |
|---------------|-----------|--------|
| 5 salamanders | ✓ | 0.0 |
| 6 crocodiles | ✓ | 0.0 |
| 7 lizards | ✓ | 0.0 |
| 8 birds | ✓ | 0.0 |
| 9 mammals | ✓ | 0.0 |
| 10 snakes | X | 7.7 |

If you wish to include taxa, then select them in the taxon list window, and choose **Include** from these same menus. Newly included taxa will always be added to the base of the tree in a ladder-like fashion.

Branch storage

If you pick up a branch with the arrow tool, the arrow in the tool palette will change to a diamond or "storage tool". If you drop the branch onto this diamond in the palette, MacClade will temporarily remember the branch. Clicking the diamond onto another branch or a tool will be equivalent to dropping the stored branch onto the other branch or tool. This can be handy for moving branches on a very large tree. You can first store a branch at one end of the tree, then scroll to a second branch far off the screen at the other end of the tree, then drop the first branch on the second.



Searching for short trees

MacClade 4's search tool looks for simple rearrangements of the tree that result in a shorter tree (i.e., a tree with a smaller treelength [[page 374](#)]; a more parsimonious tree).

If you touch the search tool on a branch, MacClade will look for rearrangements of branches above (or below, if another mode has been chosen) the branch touched that yield shorter trees. A window will appear showing the progress of the search:



You can abort searching, and show the tree obtained so far by swapping, by hitting Command-period (⌘-.) on the keyboard or by pressing the Stop button. This tool does not consider continuous characters in its searching.

This tool has multiple options, available in the tool's pop-up menu:



By default, the tool searches for rearrangements that improve treelength above the branch clicked. Choosing **Search below branch** from the tool's pop-up menu or holding down the Option key will cause MacClade to try rearrangements outside of the clade touched.

MacClade tries two types of branch rearrangements: subtree-pruning-regrafting (SPR; Swofford et al., 1996) and nearest-neighbor interchange (NNI; Swofford et al., 1996). By default, it tries all NNI rearrangements, and then, if no better tree is found, tries SPR rearrangements until it finds one that improves tree-length. After making the single SPR rearrangement, it then returns to do multiple NNI rearrangements, then one SPR, then multiple NNI, and so on. If you turn off **SPR+NNI combination search**, then MacClade only attempts NNI rearrangements.

If **Multiple Rearrangements** is checked, MacClade will only make rearrangements that improve tree-length, and it will continue making rearrangements until it cannot find any that improve the treelength. If **Multiple Rearrangements** is unchecked, MacClade will make the first rearrangement that improves the

treelength OR yields a tree of equal length, and it will only make one such rearrangement. This is useful for finding several equally optimal trees. Because the swapping algorithms visit the branches on the left side of the tree first, MacClade will preferentially make swaps on the left side.

If you choose **Search for longer trees**, MacClade will search for longer (less parsimonious) rather than shorter trees.

To help you track where branch rearrangement are taking place, a spot (•) appears on the branch each time that branch is moved. If you turn on **Show new trees during search**, however, then rather than showing a spot on the branches, the entire new tree is drawn.

If **Show rearrangements considered** is checked, then MacClade will display the branches it is considering moving as it does its calculations; those branches are marked by ?. This can be useful in teaching. However, as the search can often proceed too quickly to watch the path of rearrangements considered, you can also slow down the search by choosing **Slow search**.

MacClade is not nearly as capable as programs designed primarily to find optimal trees, such as PAUP* (Swofford, 2000) or NONA (Goloboff, 1999), and is less likely to find optimal trees. It does, however, conduct a much more thorough search than MacClade 3's search tools did. For example, for one data matrix, if one gives MacClade 3, MacClade 4, and PAUP*4 the same randomly generated tree, MacClade 4 does much better than MacClade 3 at finding the shortest tree, as indicated in the following table of one example search:

| Program | Length of trees found | Number of trees found | Time taken |
|---------------|-----------------------|-----------------------|------------|
| MacClade 3.08 | 846 | 1 | 24 s |
| MacClade 4 | 658 | 1 | 108 s |
| PAUP*4 | 654 | 69 | 6.25 s |

(For this matrix, the shortest trees are of length 653.)

Because MacClade's forté is not in finding the shortest trees, it would be most profitably used in combination with the more sophisticated tree-finding programs. MacClade could be used initially to test favorite trees, explore the data matrix, or explore different assumptions, and then the data could be sent through one of the above programs for a more powerful tree search. The user could then return to MacClade to explore the newly found trees. See [Chapter 5](#) and [Chapter 8](#) for discussion of using MacClade with other programs.

Nonetheless, MacClade can be used to search for short trees. One way to proceed is to begin with a fairly reasonable tree (perhaps based on your experience with the group), and use local branch swapping on the whole tree. Try alternating between multiple and single rearrangements (because single rearrangement searching can find equally parsimonious alternatives that may allow you to escape from a local optimum). Next, examine the character state distributions (perhaps using the character Data Boxes), to see if any other rearrangements are suggested. After you've made some moves, try the branch-swapping tools again.

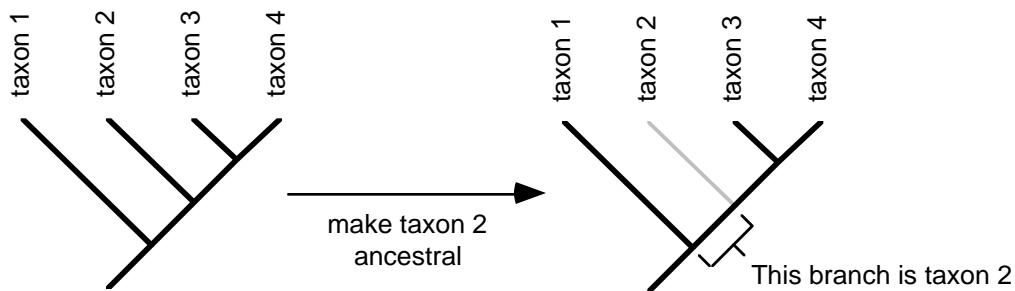
NOTE: *MacClade swaps only on dichotomous trees; thus, if your tree has polytomies, MacClade will ask you if you want to randomly resolve the polytomies into dichotomies before swapping. See [Chapter 23](#) to learn more about the random resolution.*

Making an observed taxon ancestral

To push an observed taxon downward to be an ancestor, use the make-ancestor tool. (This tool is available only if there are no characters of user-defined or Dollo types, and can be used only on a terminal branch.)

In normal mode, this tool forces the selected observed taxon to become an ancestor. The taxon is put in the place of the branch that used to be its immediate ancestor. The name of the taxon remains on top because it is easier to draw it there. From the name is a thin line pointing down to the taxon's actual position as ancestor. Of course, it doesn't make sense to use this tool if the observed taxon is a higher taxon including more than one species (e.g., a genus or family).

NOTE: An observed taxon fixed as an ancestor can have only one descendant node in MacClade. That is, you cannot place an observed taxon directly onto a branching point from which two clades or species descend.



All of the character states of the taxon are forced down onto this ancestral branch. This is useful for paleontologists, because they can see the number of extra steps in character evolution required by supposing a fossil is an ancestor. It is also useful for neontologists to determine if a species might be paraphyletic according to the tree: If the treelength goes up when it is made an ancestor, then the species has autapomorphies. (Trace All Changes can also give a summary of autapomorphies.)

Even though a taxon made ancestral has only a line coming down from the taxon's name instead of a full terminal branch, this line can still be grabbed with the mouse. However, the only thing that can be done with this line is to apply the make-terminal tool to it.

A taxon fixed as ancestor remains so fixed even when another character is traced. The ancestral status of a taxon is stored when the tree is saved to the file.

A taxon fixed as ancestor can be made terminal again by reapplying the make-ancestor tool to it. All taxa fixed as ancestors can be made terminal by choosing **All Terminal** from the **Trees** menu.

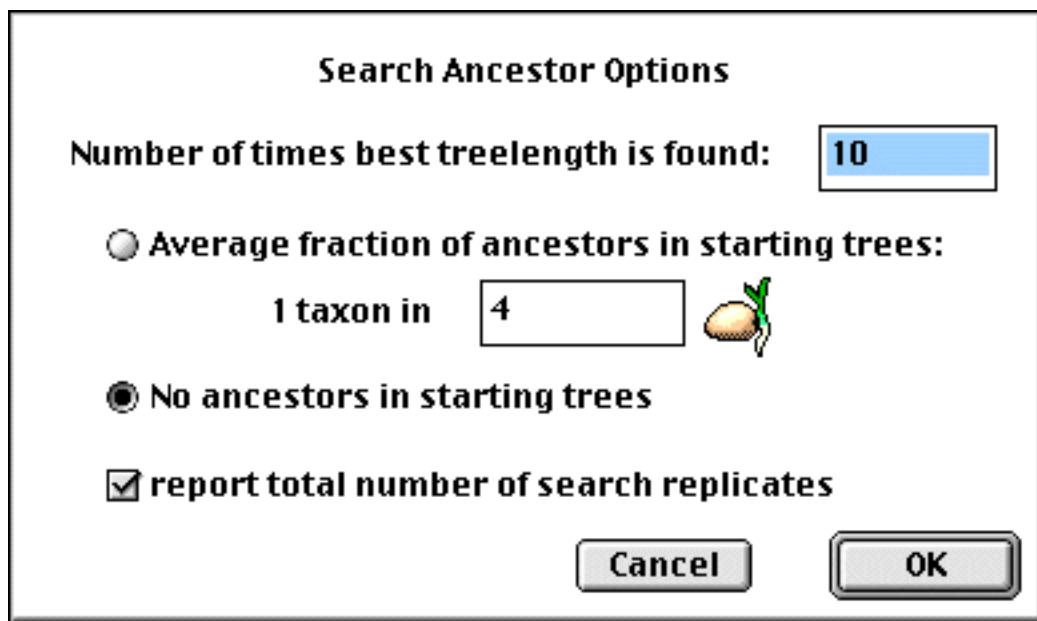
WARNING: If an observed taxon that is polymorphic for some character is forced down as an ancestor, the number of steps required is likely to be underestimated. A warning to this effect is given by MacClade in most circumstances. For example, when an observed taxon polymorphic for 0,1 is forced down as ancestor into a region of the tree with all state 0 surrounding, an extra step should be counted (gain, then loss, of 1) but isn't. The shading of branches indicating character evolution should be correct, however, except perhaps for irreversible characters. Note also that for Dollo and irreversible characters you can force the assumptions of the methods to be violated (for instance, if you make a taxon with state 1 ancestral to a clade with state 0 for an irreversible character). MacClade will not always give you a warning if you do this. Stratigraphic characters do not encounter such problems with polymorphisms.

Searching for optimal ancestors

The make-ancestor tool has another mode, available as an option in the tool's pop-up menu. In normal

mode, the tool makes a terminal taxon an ancestor, or makes a taxon designated previously as an ancestor terminal, as described in the preceding section. But if **Search Mode** is chosen, then this tool searches for the combination of terminal taxa designated as ancestors that gives the shortest treelength.

In particular, if you touch the make-ancestor tool in search mode on a branch, MacClade will examine the terminal taxa above that branch, and try various combinations of terminal taxa designated as ancestors in an attempt to find the combination that minimizes treelength. You will be presented with a dialog box in which you can set options:



MacClade proceeds by taking the tree on the screen, forcing all terminal taxa to be terminal. Depending upon the options chosen, it makes some of the terminal taxa ancestors, and then randomly chooses a terminal taxon to make ancestral or terminal. If so doing improves the treelength, then that change is accepted in the tree, and another terminal taxon is chosen randomly (without replacement) to test if the treelength would be improved if its status were reversed (terminal to ancestral, or ancestral to terminal). Once any single change of status fails to improve the treelength, the replicate is completed, the treelength is recorded, and another replicate is conducted. This is repeated until the best treelength is found at least the designated number of times (by default, 10, but you can change the number in the dialog box).

By default, the starting tree for each replicate is the tree on the screen, but with all taxa designated as terminal. If you wish each taxon in the starting tree to be ancestral with a particular probability, then choose "Average fraction of ancestors in starting trees" and modify the average fraction if desired.

Note that during searching, the topology of the tree (exclusive of the ancestral/terminal status of the taxa) is not altered.

Undoing tree manipulations

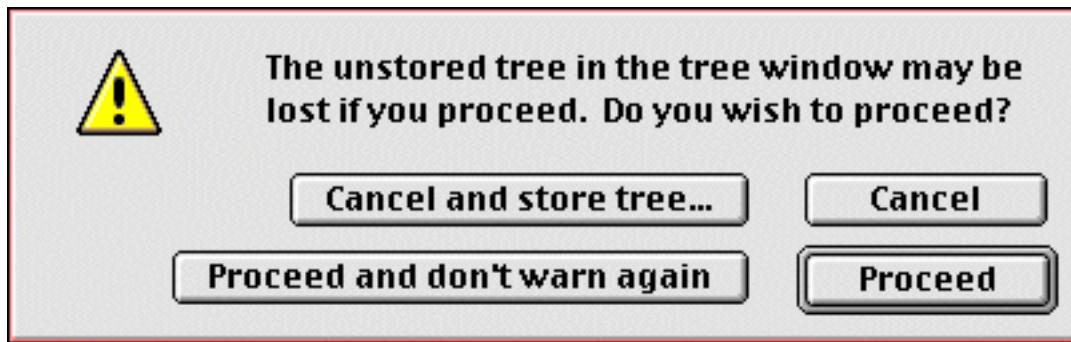
After you make a change in tree structure, MacClade remembers the previous tree, so that you can undo the change and recover the old tree. To recover the old tree, choose the **Undo** command from the **Edit** menu. You can undo tree manipulations that had a substantive effect on the tree, but some cosmetic manipulations cannot be undone by the **Undo** command. The **Undo** command will reflect which type of move is to be undone; the menu item will read "Undo Branch Move", "Undo Rerooting", "Undo Branch Swap", "Undo Get Tree", "Undo Collapse", "Undo Inclusion-Exclusion", or "Redo". Choosing **Redo** will get

back the tree you just undid.

Undo will not always restore a tree exactly to its former appearance. Some nodes may have the left-to-right order of their descendants reversed, for instance. The important aspects of tree structure will be restored, however.

Asking MacClade to warn you if the current tree will be lost

If the current tree in the tree window is unstored, and you either alter the tree or ask MacClade to display a stored tree, the unstored original tree may be lost (although, until you have made two changes, you can use Undo to recover the original tree). By default, MacClade does not warn you about the potential loss of unstored trees. However, if you wish it to warn you, choose **Warnings** from the **Trees** menu, and turn on "warn if unstored tree is about to be lost". In this mode, when an unstored tree may be lost, you will be presented with the following warning:



Tree display commands

The following commands change the appearance of the tree on the screen, but do not affect the tree in a substantive way. That is, treelength, character tracing, and other such results will not be affected. (Subsequent use of the search tools may be affected, however, because their algorithms visit one side of the tree first.) For these changes, there is no Undo command available that would automatically reverse their action.

Rotating branches

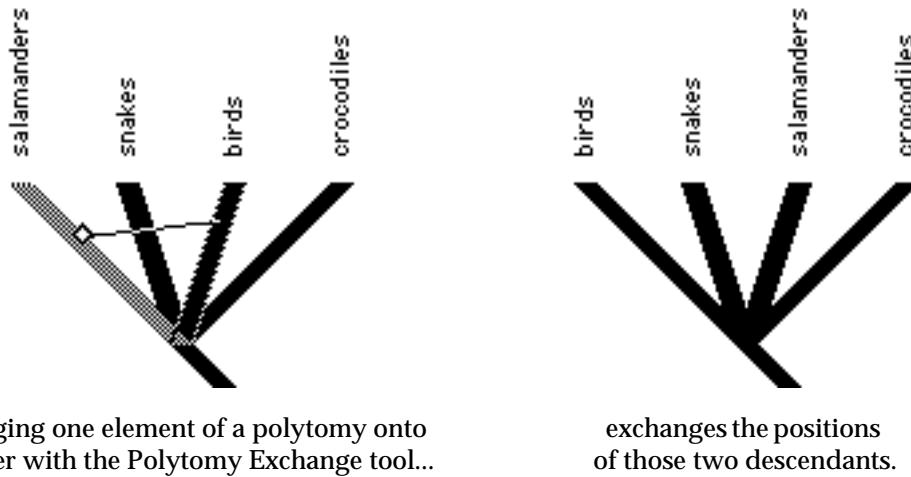
There may be times when you wish to change the order of sister groups on the screen. You can accomplish this by invoking the rotate-branches or rotate tool. For example, to reverse the position of crocodiles and birds in the following example, click the rotate tool on the branch below the clade:



Touching the rotate tool on branch... rotates the two descendants

⌚ Exchanging elements of a polytomy

The rotate-branches tool will not work on a polytomy. In order to rearrange two elements of a polytomy, use the Polytomy Exchange tool. Click on one element of the polytomy, and drop on another element of the same polytomy. The two elements will exchange places.



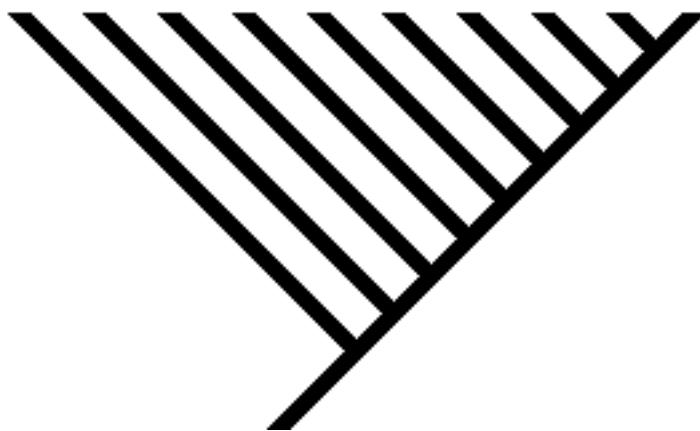
☒ Ladderizing clades

In the course of moving branches around, you might end up with a jagged clade, looking rather like a monkey-puzzle tree:

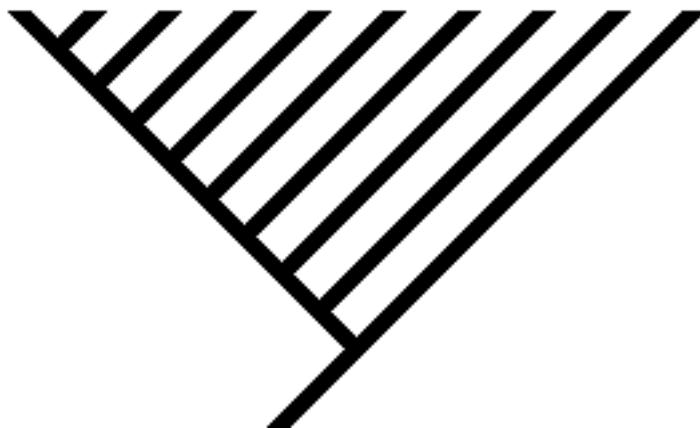


If you wish to tidy the clade up a bit by reducing the jagged aspect, you could use the rotate-branches tool to reorganize the clade. But this can be very time consuming. A quicker method is to use the lean-right tool

(☒), which automatically rotates branches to produce the arrangement that is maximally comb-like. Ladderizing the above jagged clade using the lean-right tool would yield the following arrangement:



If you hold down the Option key, or choose **Ladderize left** from the tool's pop-up menu, the lean-left tool (☒) will become available, which will ladderize a clade with the backbone of the ladder directed toward the left:



It should be emphasized that this does not change the phylogenetic structure of the tree; it only changes its appearance.

🔍 Expanding and shrinking

You can expand the tree to get a close up of a clade by clicking the magnifying-glass tool on the branch below the clade. This fills the tree window with the clade chosen. The remaining part of the tree, below and beside the displayed clade, is not visible. Although it is not visible, for the purposes of all calculations such as tracing character evolution, the remaining part of the tree is still attached to the displayed clade.

Expanding a clade does more than just allow you to look more closely at it. Some of MacClade's tree manipulation functions, such as rerooting, collapsing below, and cut below, apply to the whole tree except when a clade is expanded, in which case they are restricted to the clade. For instance, if you want to reroot a clade but leave the rest of the tree intact, expand that clade to fill the screen before using the reroot tool.

If the tree is expanded, you can shrink it back so that the whole tree fills the tree window by clicking the magnifying-glass tool on the basal branch on the screen. You can also shrink the tree by using the **Show**

All Clades command in the **Display** menu.

NOTE: Double-clicking on a branch with the arrow tool is equivalent to using the magnifying glass on that branch.

NOTE: If you hold the Option key down while you shrink the tree using the mouse, the tree shrinks only enough so that the sister group of the viewed clade is included, rather than shrinking down so that the whole tree is visible.

Display of taxon labels

You can change the display of labels used at the tips of the trees using the **Taxon Labels** dialog box in the **Display** menu. The state names of the currently traced character can be shown at the tips of the tree's branches instead of taxon names, if you so choose in the **Taxon Labels** dialog box. You can also ask MacClade to write the taxon numbers at the tips of the branches.

To change the font, size, and style of the text, choose the relevant items in the **Font**, **Size**, and **Style** sub-menus in the **Display** menu. (Although this allows you to italicize all the labels, you can individually italicize a taxon name if it is selected in the data editor and **Italics** is chosen from the **Display** menu.) These fonts can be saved as preferences for future files using the "tree window options" of the **Save Preferences** dialog box under the **File** menu (["Designating preferences" on page 116](#)). Note that the font used in the tree window does not affect the font used in tree printing or saving of graphics files.

Square or angled trees

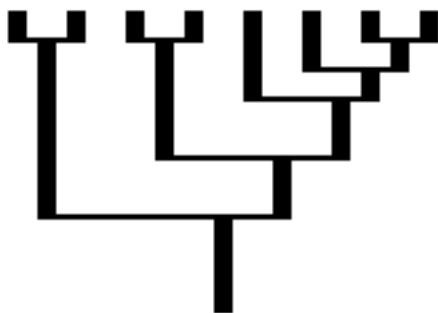
By default, MacClade draws trees on the screen with diagonal, angled branches. If you want square branches (consisting of only horizontal and vertical lines), then choose **Tree Shape & Size** in the **Display** menu. In the dialog box presented, you can choose the shape of tree, by selecting either of the three icons:



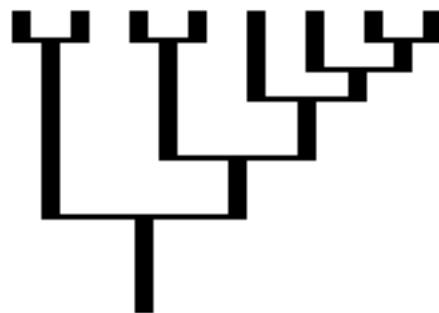
Choosing the leftmost icon will cause MacClade to draw trees with diagonal, angled branches. Choosing the middle icon will cause MacClade to draw trees with horizontal and vertical branches, and all branch tips at the same height. Choosing the rightmost icon (available as an option only when Trace All Changes is shown) will cause MacClade to draw trees with horizontal and vertical branches with branch lengths proportional to the number of reconstructed changes on the branch. If you choose the centered branches option with square branches, then MacClade will adjust the horizontal position of branch stems so that they are centered under the immediate descendants. This produces somewhat more pleasing trees but forces MacClade to redraw the entire tree more frequently. Some of these variants are shown below.



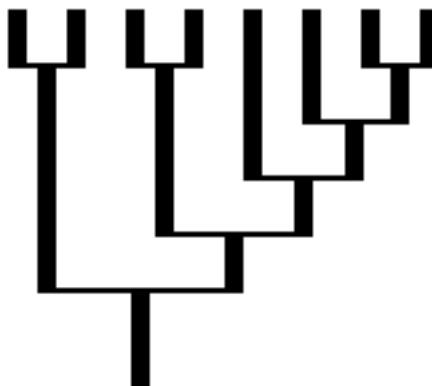
Angled tree



Square tree, uncentered branches



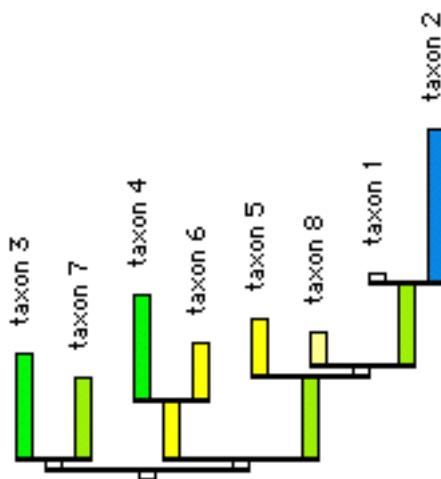
Square tree, centered branches



Square tree, centered branches, even vertical spacing

Branch lengths proportional to number of reconstructed changes

You can request that MacClade draw the length of each branch proportional to the number of changes reconstructed along the branch. To do this, first make sure **Trace All Changes** is on ([page 358](#)), then choose **Tree Shape & Size** from the Display menu. Select the third shape icon, on the right hand side, and press OK; MacClade will then draw branches with varied branch lengths.



A tree showing Trace All Changes with branch lengths proportional to number of reconstructed changes

If minimum-average-maximum changes are calculated, then MacClade will draw the branch lengths proportional to the maximum changes. (Note that to the length of each branch is added a small amount [4 pixels]. This enables even zero-length branches to be picked up and moved around.) When proportional branch lengths are shown in the tree window, MacClade will automatically adjust the vertical scale of the branches so that bars and labels on branches will be visible. However, this is not done if you have asked MacClade to weight changes by the cost of changes or character weights in the **All Changes Options** dialog box.

There are many options for display of the branches with Trace All Changes; see "[Branch display](#)" on [page 360](#).

Tree size

You can change the size of the tree in the tree window using the **Tree Shape & Size** dialog box in the **Display** menu. In it you can set a scaling factor, which will increase or decrease the tree in both horizontal and vertical directions. This scaling factor is entered in the "Tree size: times normal size" box. You can further adjust the height/width ratio of square trees by entering the height/width ratio in the " times taller than wide" box. A height/width ratio of less than 1.0 is not allowed.

Numbering branches

The **Branch Numbers** item of the **Show** submenu in the **Display** menu will draw small numbers over the branches of the tree. MacClade uses branch numbers to refer to branches or nodes when printing or saving text files. For instance, when you ask to print a Node List when a character is traced on the tree window, MacClade will list branch numbers, and beside each indicate the states reconstructed at the node at the top of the branch.

Trace labeling

Various options are available for altering the manner in which elements are traced upon the tree, using the **Trace Labeling** and **Patterns & Colors** dialog boxes in the **Display** menu. For full details, see [Chapter 18](#) and [Chapter 21](#).

Showing pictures and footnotes

To show (from the tree window) a picture that was attached to a taxon, character, or a cell in the data matrix, use the show-picture tool, as described under "[Show Picture tool" on page 446](#). To show a footnote, use the standard arrow tool, as described under "[Footnotes" on page 443](#).

Tree storage and retrieval

Trees in data files and tree files

Trees can be stored in the data file, or in an external tree file.

When you first start a data file, MacClade will presume that the trees are in the data file. If you store trees, they will be stored to the data file. If you request to get trees using **Tree List** in the **Trees** menu, MacClade will look to the data file for trees.

However, you can specify an external tree file as the source and repository of trees. Do this in the tree window using **New Tree File** or **Open Tree File** in the **Trees** menu. If you choose **Open Tree File**, a dialog box will come up allowing you either to open an existing external tree file (for instance, one saved as the result of a PAUP* run), or to create a new one by clicking the New button. Once you have opened or created an external tree file, MacClade will place any stored trees in this external file, and will get trees from there when requested. Remember that the trees will be saved to the disk only when you ask to **Save Tree File**. If you do switch to an external tree file, the trees stored in the data file will not be lost. When you close the external tree file using **Close Tree File**, tree storage and retrieval reverts to the data file.

Thus, at any one time, MacClade uses only one **tree repository**. This repository is the tree file if a tree file is open, or the data file if no tree file is open.

WARNING: *If you delete, add, or change the names of taxa in the data editor, then MacClade will automatically adjust the tree descriptions stored in the data file and any currently open external tree file. It will not, however, update the tree descriptions in any tree files that are not open when the taxon change is made, and these tree files might therefore be left with archaic taxon names that will cause problems in later reading those tree files. However, if you have tree files with archaic names, it is relatively easy to edit manually the files with a text editor to bring them into compatibility with the more newly modified data files.*

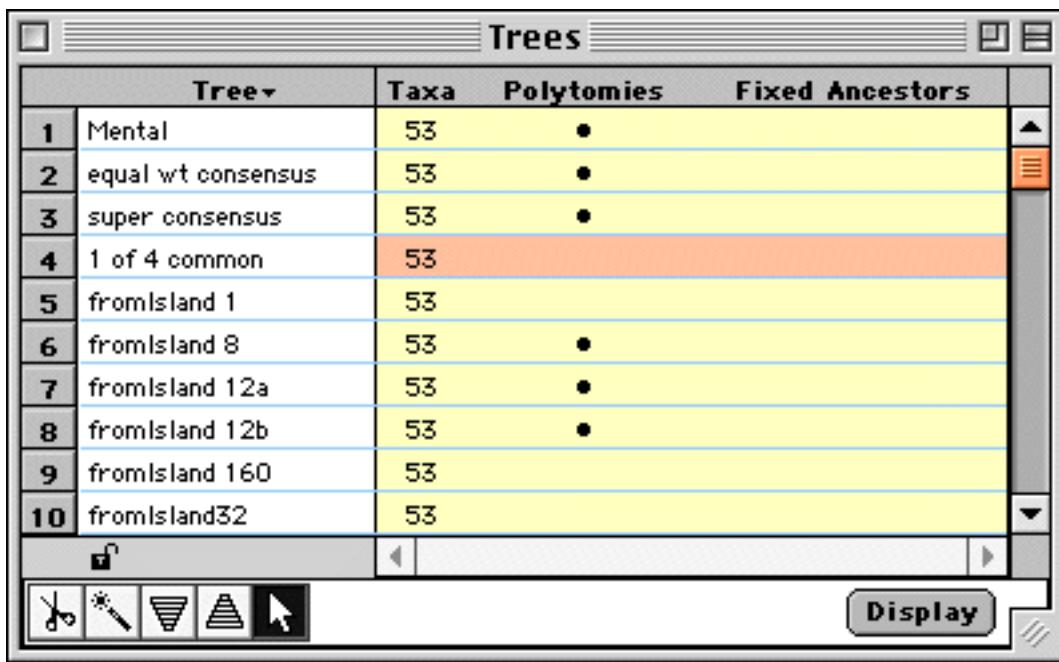
If you want to examine only some of the trees in a tree file, you can specify a range of trees for MacClade to read in the dialog box that appears when you click on the Options button in the **Open Tree File** dialog box.

MacClade can read tree files created by MacClade, PAUP 3.0 and PAUP*4, HENNIG86, NONA, and PHYLIP. If MacClade cannot detect the file format of the tree file, it will ask you to specify the kind of tree file it is.

A third file can be temporarily accessed by asking a chart to be made of some aspect of the trees in another tree file. This other tree file will be read only for the purpose of the chart; once the chart is calculated, MacClade will revert to using whichever tree repository (data file or external tree file) it had been using. See [Chapter 20](#) for more information.

The list of stored trees

To see the list of stored trees, choose **Tree List** from the **Trees** menu. You will be presented with a list window ([Chapter 10](#)) showing the stored trees in the current tree repository, as shown in the following example.



The screenshot shows a MacClade window titled 'Trees'. The main area is a table with columns: 'Tree', 'Taxa', 'Polytomies', and 'Fixed Ancestors'. The rows list various trees with their respective values. Row 4 ('1 of 4 common') has a light orange background. Row 10 ('fromIsland32') has a light yellow background. The bottom of the window features a toolbar with icons for selection, zoom, and other functions, and a 'Display' button.

| Tree | Taxa | Polytomies | Fixed Ancestors |
|----------------------|------|------------|-----------------|
| 1 Mental | 53 | • | |
| 2 equal wt consensus | 53 | • | |
| 3 super consensus | 53 | • | |
| 4 1 of 4 common | 53 | | |
| 5 fromIsland 1 | 53 | | |
| 6 fromIsland 8 | 53 | • | |
| 7 fromIsland 12a | 53 | • | |
| 8 fromIsland 12b | 53 | • | |
| 9 fromIsland 160 | 53 | | |
| 10 fromIsland32 | 53 | | |

Displaying a stored tree

To display a stored tree, open the tree list window (available by choosing **Tree List** in the **Trees** menu). You will be presented with a list of the trees stored in the current tree repository (data file or external tree file, whichever is active). Select a tree in the list (["Selecting objects" on page 163](#)), and then click on the **Display** button. Simply double-clicking on a tree's number is another way to choose it. That tree will then be displayed.

You can scroll through trees stored in the file by using the scroll bar in the tree legend window.

NOTE: *The up arrow (↑) key will take you to the next stored tree if the tree window is the frontmost window; if no stored tree has been displayed, it will take you to the first stored tree; if a stored tree was displayed, but has since been modified, then it will take you to the next stored tree beyond the one previously displayed. The down arrow (↓) key will take you to the previous tree.*

NOTE: *Double-clicking on the title bar of the tree legend will bring up the tree list window.*

Displaying predefined trees

There are two predefined trees: the default ladder and default bush (see ["Getting to the tree window" on page 311](#)). These are both available in the **Create Trees** submenu of the **Trees** menu.

Deleting, reordering, or renaming stored trees

You can delete, reorder, or rename trees using the tree list window (see [Chapter 10](#)).

Storing trees

To store a tree for later retrieval, choose **Store Tree** from the **Trees** menu. You will be presented with the tree list window, in which will appear the newly-saved tree as the last row in the list. You can then name the new stored tree. Remember, this does not save trees to disk; it only stores them in memory. To save

them to disk, you must then choose **Save File** or **Save Tree File** from the **File** menu.

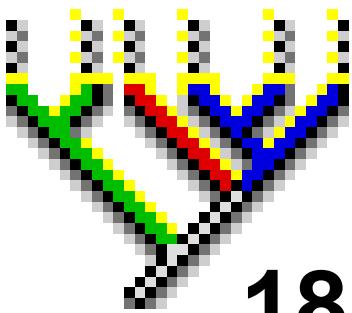
Saving external tree files

You can save a tree file to disk using the **Save Tree File** and **Save Tree File As** commands from the **Trees** menu; both of these will save tree files in NEXUS format. If you wish to save a tree file in a different format (HENNIG86 or PHYLIP), then use the **Export Tree File** menu.

Transferring trees between data files and tree files

To transfer trees from an external tree file to the data file, open the tree file, and then open the tree list window. In the tree list window, select the trees you wish to transfer, and choose **Transfer Trees** from the **Trees** menu. This will copy those trees to the data file.

To transfer trees from the data file into a new, external tree file, first make sure any external tree files are closed, and open the tree list window. In the tree list window, select the trees you wish to transfer. Then choose **Transfer Trees** from the **Trees** menu. This will create a new, unsaved external tree file, and will copy the selected trees to that tree file.



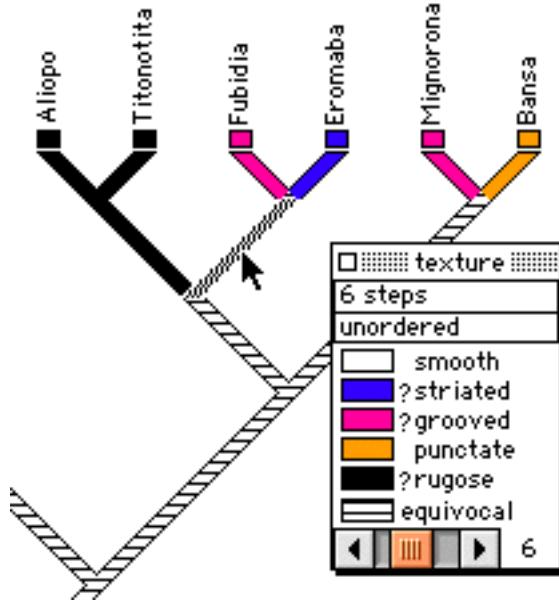
TRACING CHARACTER EVOLUTION

18

In this chapter we describe MacClade's methods for reconstructing evolutionary history of discrete-valued characters (for continuous-valued characters, see [Chapter 21](#)). The reader is referred to [Chapter 3](#) for discussion on the importance of such reconstructions, and to [Chapter 4](#) for the theory of parsimony reconstructions of character evolution.

Tracing the history of character evolution

In the tree window, MacClade will reconstruct the evolution of a character in the data matrix and display the reconstruction graphically by tracing (painting) onto the tree the inferred states of ancestors.



The tracing displays the ancestral character states that imply the smallest amount or number of character state changes. That is, the tracing shows, on each of the branches, the most parsimonious (simplest) hypothesis of ancestral states. *Do not forget that these reconstructions are estimates based on parsimony considerations; they are not observed facts.* To be precise, each tree branch is shaded to indicate what states are most parsimoniously placed at the node terminating the branch. The set of states that can be parsimoniously placed at a node is the node's MPR set, as described in [Chapter 4](#) (see also Swofford and Maddison, 1987). Each state in the MPR set occurs in at least one most parsimonious reconstruction (MPR) of character evolution. A box that appears just under each terminal taxon's name shows the state(s) observed for that taxon. If the state is unknown for a taxon (coded in the data matrix as missing), no box appears. A legend is provided on the right of the screen showing the relationship between pattern, shade, or color and the state.

Which states are reconstructed at the various ancestral branches depends on the shape of the tree, the

observed states in the terminal taxa, and various background assumptions concerning the nature of changes from one state to another in the character. In particular, each character is assigned to a transformation type (e.g., unordered, ordered, irreversible) that indicates how many steps are counted for a change from each state to each other state. Thus with an unordered character, state i can change to state j in 1 step, whereas with an ordered character state i can transform to j in $|i - j|$ steps. The character tracing displays the reconstruction that is most parsimonious in the sense that it requires the fewest steps so counted.

[Chapter 3](#) and [Chapter 4](#) should be consulted for more information on the principles and assumptions underlying MacClade's ancestral state reconstructions.

For polytomous trees, MacClade allows two alternative interpretations of the polytomies (soft vs. hard polytomies) that affect the reconstruction of character evolution. Polytomies are difficult to deal with, and the reader should consult W. Maddison (1989) and the discussions of polytomies in Chapters [4](#), [17](#), and [20](#).

EXAMPLE: *Donoghue (1989) presents an analysis of the evolution of dioecy and dispersal mechanisms among the seed plants. With the aid of MacClade 2.1 he reconstructed, as one of several possibilities, the scenario presented in the file "Dioecy Example" in the Morphological examples folder. Donoghue sought a correlation between this character's evolution and the evolution of dioecy. Using W. Maddison's (1990) test, he found that reconstructed gains of dioecy may or may not be significantly concentrated in the phylogenetic context of fleshy fruit (= animal dispersal?), depending upon just how history is interpreted. This illustrates the important point that the sequence of events (which came first, which later) is critical to know if we are to distinguish various evolutionary processes.*

Choosing a character to trace

There are several ways to trace a character:

1. Choose **Trace Character** from the **Trace** menu. This will display the last character traced, or the first included character if none has yet been traced.
2. If Data Boxes are visible ([page 357](#)), then double-click on the line of boxes of the character to be traced.
3. If the character list window is visible, then double-click on the row of the character to be traced, or select a row and press Trace.
4. If Trace All Changes or All States are being shown, and you have queried about the changes along or states at a branch using the branch-information tool, then double-click on a line in the window listing changes or states to turn off the summary mode and trace that character.
5. If a character is already traced, then the **Go To** submenu in the **Edit** menu allows you to type in the number of another character you wish traced.

You can also temporarily trace a character that is indicated by a Trace All Changes bar by using the Temporary Trace tool (["Tracing a character shown in Trace All Changes" on page 368](#)).

To revert back to the last character traced, choose **Previous Traced** from the **Trace** menu.

NOTE: The right arrow (\rightarrow) key will ask MacClade to trace the next included character, the left (\leftarrow) arrow key the previous included character.

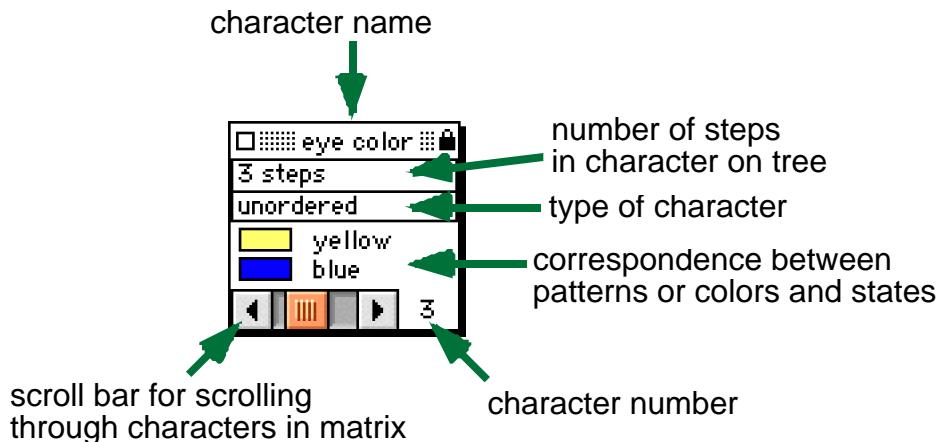
EXAMPLE: In Donoghue's (1989) example of the evolution of dioecy and dispersal in seed plants described above, you could trace the character *Dioecy*, then trace the character *Propagules*. By selecting **Previous Traced**, *Dioecy* will be traced again. You can therefore continue to choose **Previous Traced** again and again in order to alternate back and forth between the two characters. This method allows close examination of any correlations in the reconstructed evo-



lution of the two characters.

The character legend

If a character is traced, you will see a legend in the lower-left corner of the tree window, which will look something like this:



The number of evolutionary steps required in the character by the tree is shown below the character name. The number shown is simply the number of steps on the displayed (usually, the most parsimonious) ancestral state reconstruction according to the character's transformation type; it is not multiplied by the character weight. Thus if there are characters weighted differently from 1, when you add all the character steps you may not get the treelength. (Treelength is discussed in [Chapter 19](#).) The number of steps is not necessarily the number of changes of state on the tree, for example, if some changes are weighted, for instance by a step matrix. An asterisk (*) appears beside the number of steps when the character is of user-defined type and has a terminal taxon polymorphic for more than two states, signifying that MacClade's algorithms do not accurately count steps or reconstruct ancestral states in such a circumstance (see [Chapter 4](#)).

A series of smaller, shaded boxes in the character legend show the patterns or colors used on the tree. The first several boxes show the patterns or colors indicating the various character states. The lowest-valued state in the character (e.g., state 0) is shown at the top of the list, with higher-valued states below it. If you have named the states in the data file, then names of the states will appear beside the appropriate boxes (unless you request in the **Trace Labeling** dialog box in the **Display** menu that only symbols be used); if not, only the state symbols 0, 1, 2, or whatever, will appear.

The character legend is by default attached to the lower-right corner of the tree window, but you can unlock this attachment by using the **Lock Legends** item in the **Display** menu. This frees both the character legend and the tree legend to be moved independently of the tree window. When the character legend is unlocked, it will also widen to show any long character state names.

NOTE: Double-clicking on the title bar of the character legend will bring up the character list window, in which you can choose a new character to trace.

Interpreting the character tracing

The shading of the branches indicates the reconstruction of ancestral states. It is important that you know how to interpret what states are reconstructed, and exactly to what points in the tree the reconstructions

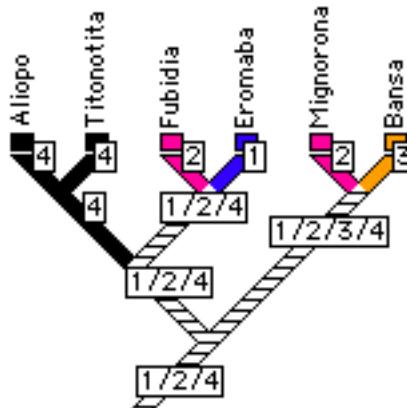
refer. The reconstruction of ancestral states determines what states are most parsimoniously placed at the branch-points or *nodes* of the tree, as defined in [Chapter 3](#). However, as explained below, for graphical convenience MacClade shades the entire branch (node plus internode below it). Users must be aware that when MacClade shades an entire branch, and when this manual refers to "state reconstructed at a branch", what is actually meant is the reconstructed state at the node at the top end of the branch.

Determining the states reconstructed at a branch

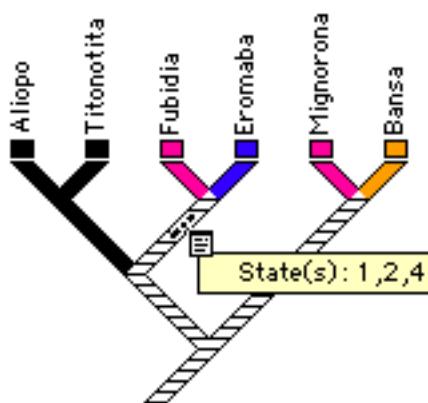
In general you can look to the character legend to see which shades correspond to which states, but the shades may be difficult to distinguish, or the character tracing may have the "equivocal", "uncertain" or "polymorphic" shadings. The equivocal pattern (■■■) is used to indicate that more than one state can be equally parsimoniously reconstructed at a branch. For stratigraphic characters, a special pattern (|||||) is used instead of equivocal, uncertain, or polymorphic shadings (see [Chapter 4](#)).

In order to determine exactly which states are reconstructed at a branch, you can use one of four methods:

1. If you place the cursor over top of the branch, asterisks or question marks appear in the character legend beside each of the states reconstructed at the branch, as shown at the start of this chapter.
2. If you ask to use text labels to indicate states (see the **Trace Labeling** dialog box in the **Display** menu), MacClade will place a label on each branch with a list of its reconstructed states:



3. If you touch a branch with the branch-information tool (📋), MacClade will momentarily show a list of the reconstructed set of states:



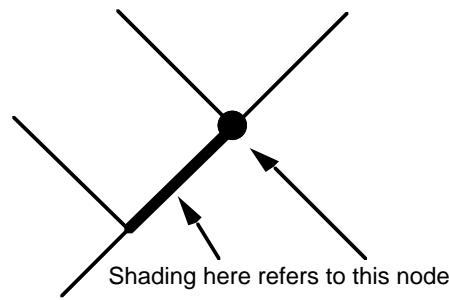
4. If you use the **Node Lists** item in the **Print Other** or **Save Text File** submenus, a list of the states reconstructed at each of the branches can be printed or saved to a text file.

Boxes for terminal taxa

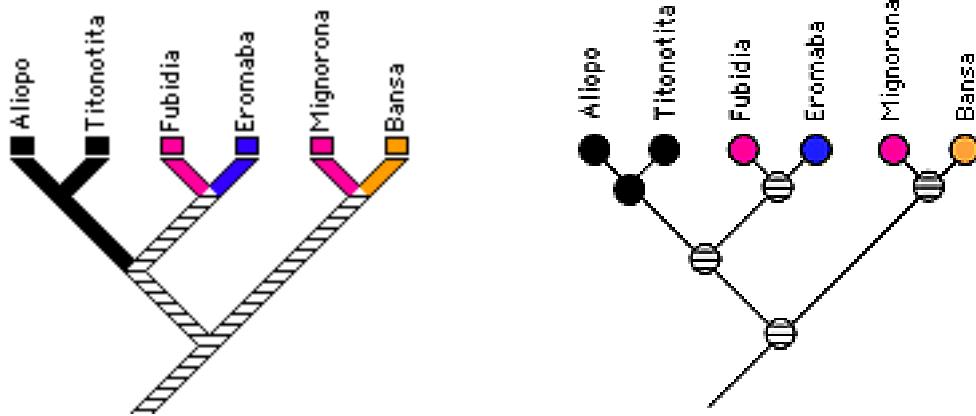
The box that appears just under each terminal taxon's name shows the state(s) observed in the taxon. If only a single state is observed in the taxon, then the pattern or color of the box will be that of the state. If the state is unknown for a taxon (coded in the data matrix as missing), or is coded as a gap, no box will appear. If a terminal taxon is polymorphic (possesses more than one state), the box and the branch below it will be shaded by a pattern designating polymorphism (▨▨▨). If there is partial uncertainty about the states of a terminal taxon (which you indicate by the use of the OR separator in the data editor) the box will be shaded by the pattern designating uncertainty (▨▨▨▨).

States of nonterminal branches

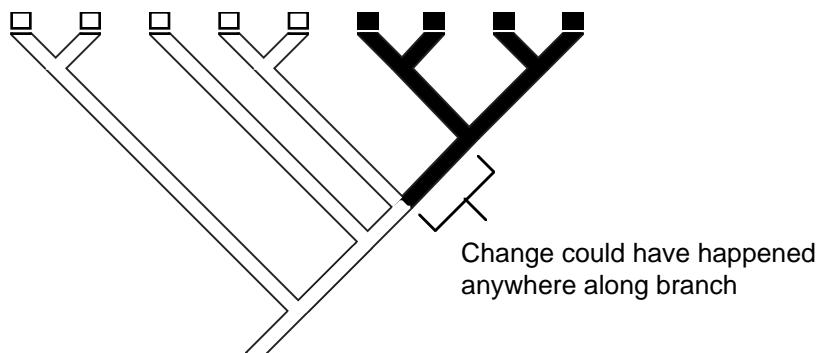
When a branch is shaded to indicate a particular state, it is important to realize to which part of the tree this reconstructed state refers. On a dichotomous tree, the branch shading is best thought of as indicating the reconstructed state at the branch point (node) at the top of the branch, as shown below.



Thus, the following two figures are equivalent:

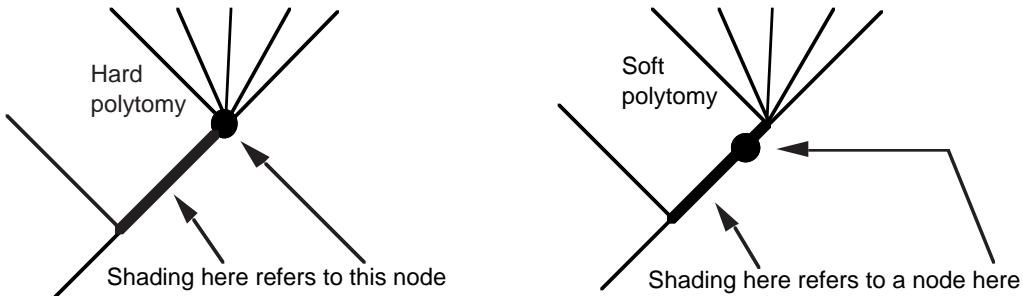


For instance, consider the basal branch of a clade that uniformly has state 1 whereas the rest of the tree has state 0:



The basal branch point of this clade is reconstructed as having state 1; the node just below this is reconstructed as having state 0. The change from state 0 to 1 might have occurred anywhere along the branch connecting the two nodes, as long as 1 is achieved before the basal node of the clade. MacClade, however, by convention shades the entire basal branch of the clade by state 1, from just above the node below it, up to and including the basal node of the clade.

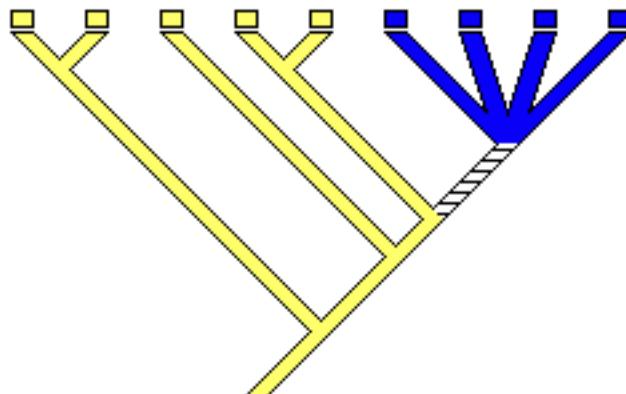
When the node at the top of a branch is polytomous, the interpretation of the branch's shading depends on whether the polytomy is assumed to represent uncertainty (soft polytomy) or multiple speciation (hard polytomy). If it is assumed to represent multiple speciation, then the comments above apply: the shading of the branch below the polytomy indicates the reconstructed state at that polytymous node, that is, at the most recent common ancestor of the clade just above branch.



If the polytomy represents uncertainty, however, then MacClade attempts to reconstruct character evolution as if the polytomy were resolved dichotomously in the way most favorable for the character. In such a dichotomous resolution, the polytomy would be broken into two to many dichotomous nodes. One might expect that the basal branch of the polytomy in MacClade would be shaded according to the states at the most basal of these dichotomous nodes, but the states at this node can be difficult to calculate and to interpret, and instead MacClade invents an extra node just between the most basal dichotomous node resolved out of the polytomy, and the node just ancestral to the polytomy. It is to this intercalated node that the shading of the basal branch of a soft polytomy refers (figure above, right).

MacClade uses this convention because the context of the basal node varies with different resolutions of the phylogeny, and this can be difficult to deal with. As there can be changes between that extra node and

the most basal node of the clade above, the shading of the basal branch therefore does not necessarily indicate the ancestral state of the clade. This can lead to puzzling results, such as that shown below.



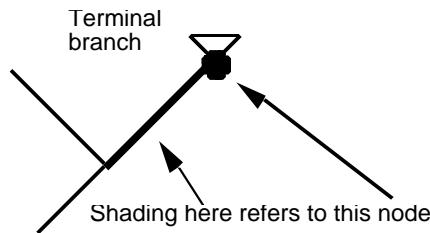
The ancestral branch of the black clade, it would seem, should be reconstructed as being black. But because with a soft polytomy the shading of the branch refers to a node just below the most recent common ancestor of the black clade, and this node might be either just before or just after the change to black, it is shaded equivocal.

States at terminal branches

The shading of a terminal branch generally indicates the observed state in the corresponding terminal taxon. However, exceptions occur when the terminal taxon has uncertain state, or is polymorphic, or when the terminal branch's state is fixed.

If the terminal taxon's state is uncertain, then MacClade assigns to the terminal branch a state that represents the most parsimonious choice, among those allowed by the uncertainty, for the taxon's state.

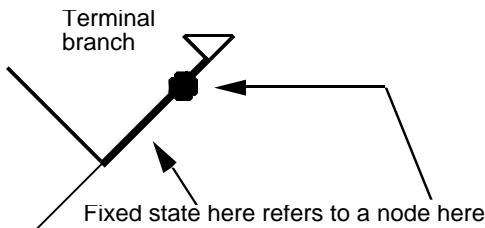
If the terminal taxon is polymorphic, then MacClade assigns a state to the terminal branch that represents the most parsimonious estimate for *the ancestral state of the terminal taxon*. That is, MacClade treats the terminal taxon as a small clade whose internal structure is unresolved, as shown below, and on the basis of the reconstructed states at nearby nodes, reconstructs what would have been the most parsimonious estimate of the state at the most recent common ancestor of the terminal taxon.



That MacClade treats terminal taxa as unresolved clades is appropriate when the terminal taxa are genera or families, but when they are species, objections may be raised to this practice. For instance, we don't expect the internal structure of a species to be purely branching. However, there seems little else we can do, given the lack of algorithms available to deal with reticulations and ancestral polymorphisms (see the discussion in [Chapter 4](#)). Note, however, that terminal polymorphisms can yield certain difficulties (see [page 43](#), and Nixon and Davis, 1991).

The interpretation of a terminal branch's shading changes, however, when the user fixes the state at the ter-

terminal branch, as when using the paintbrush tool. When a terminal branch's state is fixed, it is treated as fixing a node just below the most recent common ancestor of the terminal taxon.



This is done because of computational difficulties that arise when the fixing is considered as at the most recent common ancestor of the internal components of the terminal taxon. For instance, if the fixing is considered as at the most recent common ancestor of the components of the taxon, and the terminal taxon had state 1 and the terminal branch is fixed to 0, would we count 2 steps, one up the left and one up the right branches in the clade of the taxon's internal components? Note that MacClade 4's convention regarding fixed states differs from that in MacClade 2.1 and 1.0, in which fixing a terminal state was treated as if replacing the observed state. (See discussion of fixing the state at a branch, later in this chapter.)

A reminder about soft polytomies

Remember from the discussion in [Chapter 4](#) that a soft polytomy represents ambiguity among alternative dichotomous resolutions. MacClade's algorithms find the reconstruction of character evolution on the resolution of the polytomy that is most favorable (parsimonious) to the character. There are various problems with such an approach ([Chapter 4](#); W. Maddison, 1989), and you should avoid reconstructing character evolution on polytomous trees. Instead, examine alternative dichotomous resolutions. Be especially careful with polytomous consensus trees.

Manipulating ancestral states

Manually assigning ancestral states ("fixing states") of a branch can be useful in a number of contexts. For example, you may want to see how much it would cost in number of steps to have an alternative state in a particular ancestor. This is also useful if the initial shading of the tree had equivocal areas (where more than one state could have been placed at a branch with equal parsimony). You can then see one of the reconstructions by fixing the branch to one of these equally parsimonious alternatives (for instance, the alternative least favorable to any evolutionary hypothesis you may be testing), after which MacClade will trace character evolution in the rest of the tree given this fixing.

For example, in a study of constraints on the evolution of spider chromosomes, one of us (WPM) traced the evolution of XXO and XXXY sex chromosome systems for a genus of jumping spiders on a tree provided by the work of Griswold (1987). On this tree, MacClade's tracing showed five or six independent derivations of the XXXY system. But two questions remained: How good is the evidence that there were independent derivations (instead of a single gain and multiple losses)? And, how much does this conclusion depend on the details of the tree? By fixing the states at branches, an alternative tracing on the same tree was made in which the XXXY was gained once and multiply lost. MacClade showed that this alternative was not supported, requiring seven more chromosome fusions than the one with multiple derivations. But, on another tree differing by only a few branch moves, the difference between the multiple-gain and multiple-loss hypotheses was reduced to only one fusion.



Fixing the state of a branch

Initially, when MacClade traces a character on a tree, each branch will be shaded to indicate the states most parsimoniously placed on the branch. If there is more than one state that can be placed parsimoniously on

the branch, then you can resolve the ambiguity by fixing the branch's state to one of these parsimonious states, and then examining character evolution with the ambiguity at that branch resolved. You can also examine alternative pathways of character evolution other than the most parsimonious, by fixing the state of a selected branch to any other desired state.

Fixing a state at a branch can be done in two ways: (1) by selecting the branch, and then dropping the diamond cursor over the top of one of the shaded boxes in the right-hand character legend, which indicate the shading used for a character state, or (2) by touching the paintbrush tool on a state in the character legend to fill the brush with that state, and then touching those branches of the tree you wish to fix with that state. Any fixed branch is then forced to have that state regardless of what it costs in terms of number of steps. The surrounding branches will be reshaded as appropriate, and the number of the steps will be recalculated, given the fixing of the branch.

If character states are indicated by colors or shades of gray, then a "full" paintbrush will be the color of the state it contains; if character states are indicated by patterns, a full paintbrush will be black. An empty paintbrush is always white.

If any branch's state is fixed in the traced character,  appears beside the message box in the lower-left side of the tree window. A small paintbrush appears beside each branch whose state is fixed (unless this has been turned off in the **Trace Labeling** dialog box available under the **Display Menu**).

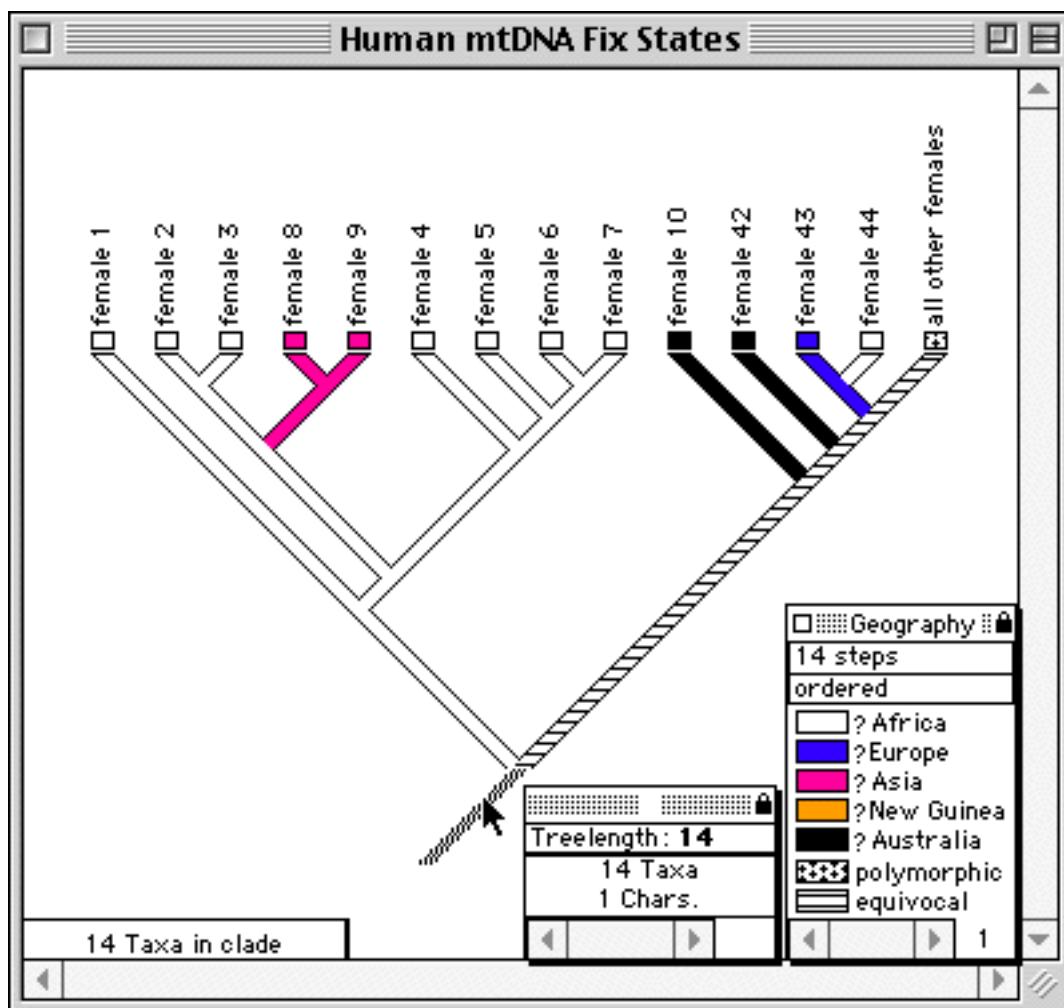
WARNING: *Fixing the state at a branch affects only the character traced, and then only the tracing pictured on screen and one chart whose results are derived from it. The only chart that is affected by fixing states at branches is the chart of State Changes & Stasis that focuses only on the traced character on the current tree. All other charts, including the Character Steps/etc. and Treelength charts, ignore the fixing and use the original, unfixed reconstruction of the character's evolution. Trace All Changes, Trace All States, and search tools likewise use the original, unfixed reconstruction.*

WARNING: *If you fix the state of a terminal branch to a state other than that observed for its terminal taxon, you do not change the taxon's state in the data matrix. Rather, MacClade supposes that you are fixing the state of a just-sub-terminal node (see [page 349](#)). Thus if the taxon had been observed to have state 1, and the terminal branch was fixed to state 0, one step would be counted to go from 0 to 1 in the terminal taxon, if the character was unordered. This convention differs from that in MacClade 2.1.*

WARNING: *For characters of some types, such as Dollo and irreversible, you can force the assumptions of the methods to be violated (for instance, you can fix state 1 at a branch below a clade with state 0 for an irreversible character). MacClade will not always give you a warning if you do this.*



EXAMPLE: Open the file "Human mtDNA Fix States". The tree you see is among the most parsimonious based on the data on human mitochondrial DNA evolution of Cann et al. (1987); see D. Maddison (1991b) for details on the derivation of the tree. The rightmost branch is labeled "all other females", and represents a clade of 121 individuals:



Move the cursor over the root branch, and note that question marks show up beside each geographic region in the character legend. This indicates that any of the five geographic regions is an equally parsimonious home of human mtDNA according to this tree. To show that this conclusion is independent of the ancestral state of the clade of 121 individuals, and thus independent of the relationships and states within the clade, you can use MacClade's paintbrush tool. First, fix the state of the branch labeled "all other females" to "Africa". Note that the root state is still fully equivocal. Now fix that branch to "Europe", then "Asia", and so on, each time noting that the root state is fully equivocal.



Examining the cost of alternative fixings of a branch

If you wish to see quickly how the number of steps in the current character varies depending upon the state assigned to a particular branch, choose **Try all states on branch** from the paintbrush tool's pop-up menu (or hold down the Option key), and click the paintbrush tool on the branch. Hold the mouse button down, and a small rectangle will appear, listing for each state the number of steps in the character if that branch were fixed to that state.

You can use this, for example, to see how much less parsimonious other assignments of ancestral state would be.



Unfixing state of a branch

To unfix the state of a single branch, choose the unfix tool (the paintbrush under water, or turpentine) and touch the branch. To unfix states of *all* branches whose states have been fixed, choose **Unfix All** from the **Trace** menu, or double click on the unfix tool's icon in the palette, or hold down the Option key and double-click on the fixed branch icon (☒) in the message box in the lower left (see "[Message box" on page 312\).](#)

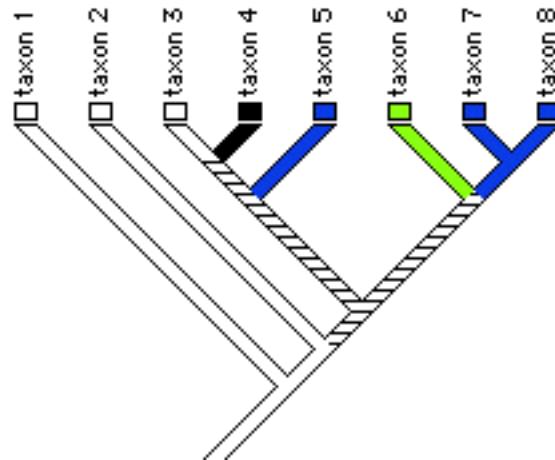
The state of a fixed branch is unfixed automatically if another branch is moved onto it, or if a different character is traced, or if the character trace is turned off.

Seeing which branches are fixed

When "labeling fixed branches" is enabled using the **Trace Labeling** dialog box in the **Display** menu, MacClade will place the end of a paintbrush (☒) beside branches whose states have been fixed. You can also see which branches have their character states fixed by touching on the fixed branch icon (☒) in the message box. Those branches whose states have been fixed will be highlighted.

Resolving equivocal tracings

MacClade's default is to indicate *all* states that can be most parsimoniously assigned to a branch. Often there are several equally parsimonious assignments, thus yielding ambiguity. This is indicated in MacClade by the presence of some branches shaded with a horizontally striped pattern, called the equivocal pattern. For the unordered character shown traced onto the tree below, there are four branches shaded with the equivocal pattern:



As noted in [Chapter 4](#), there are six equally parsimonious ways to resolve this ambiguity in this example; that is, there are six most parsimonious reconstructions (MPRs) of character evolution for this example. You can ask MacClade how many parsimonious reconstructions there are by choosing **Number of MPRs** from the **Trace** menu, or by choosing **Number of MPRs** from the **Show MPRs** menu in Show MPRs mode ([page 354](#)).



EXAMPLE: The character presented in the file "Many MPRs" has over 20,600,000,000 reconstructions on the tree!

Several methods can be used to discover and explore these alternative reconstructions.

Using the paintbrush

The paintbrush ([page 350](#)) can be used as described above to fix the state at an equivocal branch to one of the equally parsimonious assignments to that branch. The ambiguity at that branch will then be resolved, and MacClade will retrace the character assuming that the state at this branch is as fixed. Any other branches remaining equivocal can also have their states similarly fixed. In this way you can obtain one of the most parsimonious reconstructions of character evolution with no ambiguities. Although such a method may make you intimately familiar with the alternatives, it can also be tedious. MacClade therefore has some features that resolve ambiguity automatically.

Reversals and parallelisms: ACCTRAN and DELTRAN

For characters of unordered and ordered type, MacClade can automatically resolve ambiguities in character tracings so as to choose the assignments that delay (DELTRAN) or accelerate (ACCTRAN) transformations (see Swofford and Maddison, 1987). ACCTRAN and DELTRAN are described, with warnings about their use, in [Chapter 4](#).

MacClade will show you the ACCTRAN or DELTRAN reconstructions if you select these options in the **Resolving Options** dialog box available in the **Trace** menu. ACCTRAN and DELTRAN apply only to the character tracing shown on the screen and anything derived from it. These options do not apply to the Trace All Changes and Trace All States displays, nor to most of the other reconstructions of changes in the charts.

WARNING: *ACCTRAN and DELTRAN apply only to the character traced, and then only to the tracing pictured on screen and one chart whose results are derived from it. The only chart affected by ACCTRAN and DELTRAN is the chart of State Changes & Stasis that focuses only on the traced character on the current tree. All other charts and Trace All Changes and Trace All States ignore the ACCTRAN/DELTRAN option. Apart from the tracing of a character on the tree on screen, ACCTRAN/DELTRAN does apply in the one other context in which single characters are graphically traced on trees: in printing. For instance, when the Print All Characters option is selected under Print Tree, the ACCTRAN/DELTRAN option applies to all of the characters' tracings.*

Show MPRs: All parsimonious reconstructions of character evolution

For characters of all transformation types MacClade can display, one after another, all equally parsimonious resolutions of the ambiguity in the character tracing. This is accomplished with the **Show MPRs Mode**, in the **Trace** menu. (This mode was called "Equivocal Cycling" in MacClade 3.) When this command is invoked, MacClade will give one of the possible reconstructions of the ancestral states of the character traced. You can then step through the other reconstructions using the **Next MPR** command in the **Show MPRs** menu. The Show MPRs mode does respect the fixing of states at branches, and so you can ask, "What are all the parsimonious reconstructions of ancestral states, given that these branches are fixed to have these states?".

While Show MPRs mode is active on the traced character, most of MacClade's menus disappear and the other windows become inactive. Show MPRs is therefore like a locked mode. You cannot access most of MacClade's features until you turn it off, or until Show MPRs mode goes past the last reconstruction.

WARNING: *This feature can be used only with dichotomous trees without observed taxa fixed as ancestors. It cannot be used if ACCTRAN or DELTRAN options are in effect. It cannot be used on stratigraphic characters.*

MacClade, in stepping through reconstructions, always follows a consistent order. It first sets the root state to the state with the lowest state number among those that are parsimonious at the root. For information about state numbers, see "[State numbers vs. symbols vs. names](#)" on page 198. It then sets the nodes higher in the tree sequentially to the lowest most-parsimonious state given what has already been set lower down in the tree. This yields the first reconstruction. Subsequent reconstructions are found by incrementing the

states assigned to nodes, somewhat like counting, so that the terminal nodes are like the 1s digits and change most quickly, whereas the lower nodes are like the thousand's or million's digits, which change most slowly. All the while MacClade makes sure that only parsimonious reconstructions are allowed.

To go to a particular reconstruction, use **Go to** in the **Show MPRs** menu. (The **Show MPRs** menu appears only after you have chosen **Show MPRs Mode** from the **Trace** menu.) If you ask to go to the 200th reconstruction and there are only 50 reconstructions, MacClade will show you the 50th reconstruction.

When a clade is expanded to fill the screen using the magnifying-glass tool, **Show MPRs Mode** used on the character tracing on screen does not go through all character reconstructions over the whole tree. Instead, it leaves intact ambiguity in the character reconstruction below the clade, and examines alternative reconstructions only within the clade. For this reason, you may find that MacClade indicates that there are numerous different reconstructions in a character when the whole tree is shown, but when a clade is expanded, MacClade may indicate there are only a few reconstructions.

To turn off stepping through MPRs, choose **Turn Show MPRs Off** from the **Show MPRs** menu. Show MPRs mode will automatically be turned off after the last reconstruction.

All equally parsimonious reconstructions can be printed in sequence using the **Print Tree** dialog box, if the **Print All Reconstructions** menu item of the **Options** pop-up menu is checked.

MPR Calculation Modes

If MacClade 3 needed to perform a calculation that involved multiple MPRs, such as calculating the number of MPRs or calculating the average branch length over MPRs, it had to explicitly build and examine each MPR in a process that was called Equivocal Cycling (Maddison and Maddison, 1992). As described in [Chapter 4](#), MacClade 4 has alternative algorithms that can be much faster, as they don't need to explicitly examine each MPR.

By default, MacClade 4 uses the new method. However, if you wish to use the old method, the **MPR Calculation Modes** dialog box (**Trace** menu) allows you to switch to either the old calculation method ("explicitly build and examine each MPR") or to a mode in which both methods are used and their results compared. The latter was added for initial testing of the new routines, and was left in in the event that some particular result needs to be double-checked.

Display of character tracing

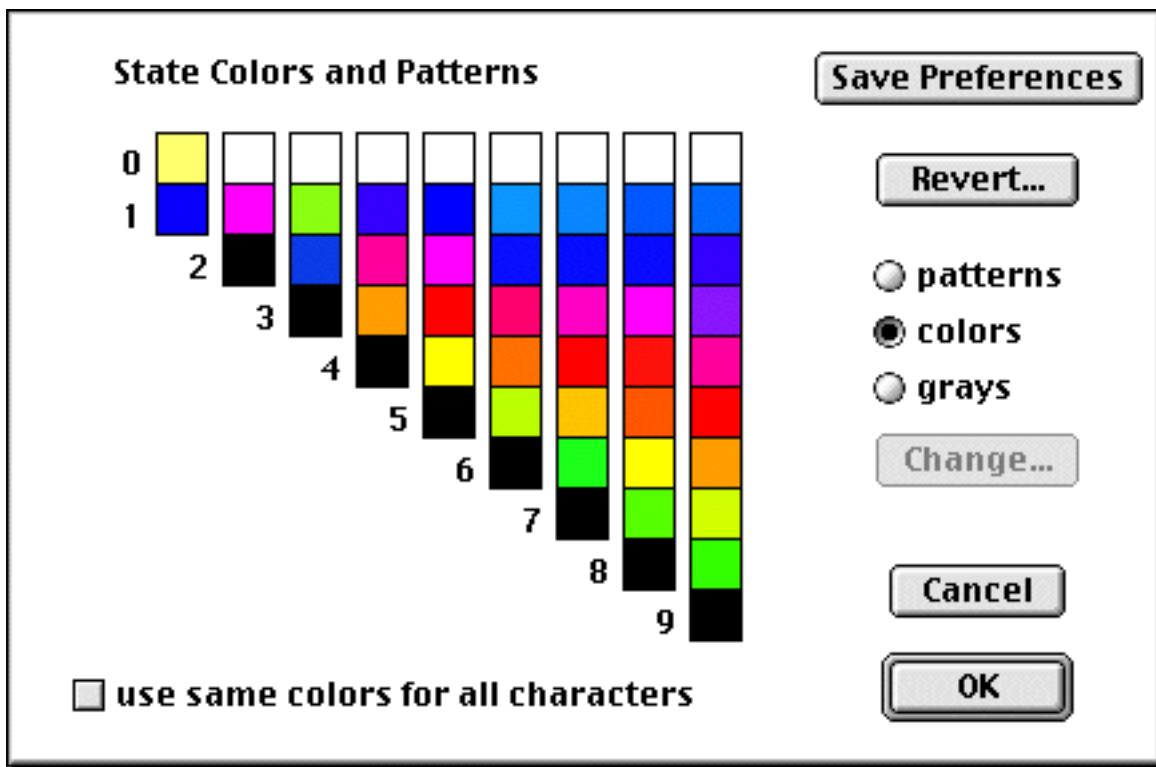
Changing patterns or shades or colors

You can change the default patterns or shades of gray or colors of traced characters, or the shadings used in the Trace All Changes mode, by double-clicking on the appropriate state box in the character legend. A dialog box will appear that lets you select a new pattern or color for that state.

For traced characters of standard or extended standard format data files, you will notice that MacClade uses the same two colors or patterns for all 0–1 two-state characters (e.g., 0 = white and 1 = black). MacClade decides what shades to use by the character's maximum observed state. All characters with maximum state 2 (or symbol corresponding to state number 2; see [page 198](#)) would have the same shades for states 0, 1, and 2. Thus, if you change the shades for one character, you will be simultaneously changing the shades for other characters.

To change the shadings for several states all at once, or to switch between patterns, colors, or grays, you can choose **Patterns** or **Patterns & Colors** from the **Display** menu. (A character must be traced or Trace All

Changes must be shown before you can change the branch shading.) A dialog box similar to this will appear:



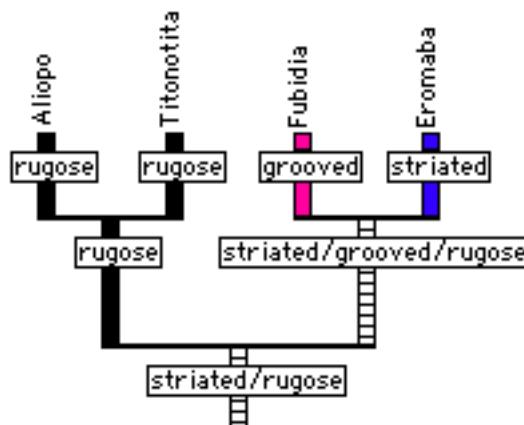
The first column on the left shows the patterns used for a character whose highest state is 1, the next column a character whose highest state is 2, and so on. To change a color, either select one of the squares in a column, and press Change, or simply double-click on the square. A dialog box will appear in which you can select a new color for that square. You can switch to displaying shading by patterns or by shades of gray using the radio buttons at right. If your Macintosh cannot display color, then your only option will be to display shades of gray or patterns. If you set it to display patterns, then you can change patterns in a similar manner to changing colors. Your chosen "palettes" can be saved to disk using the Save Preferences button.

If you check "use same colors for all characters" in this dialog box, then MacClade will use the rightmost column as the palette for all characters, independent of the highest state of each character.

NOTE: Patterns used by MacClade are stored in the resource fork of MacClade (resource type PAT#, resource id 201 and 202). You can edit them using a resource editor such as Apple's ResEdit™. **Do this on a back-up copy of MacClade, and only with caution.**

Labeling branches

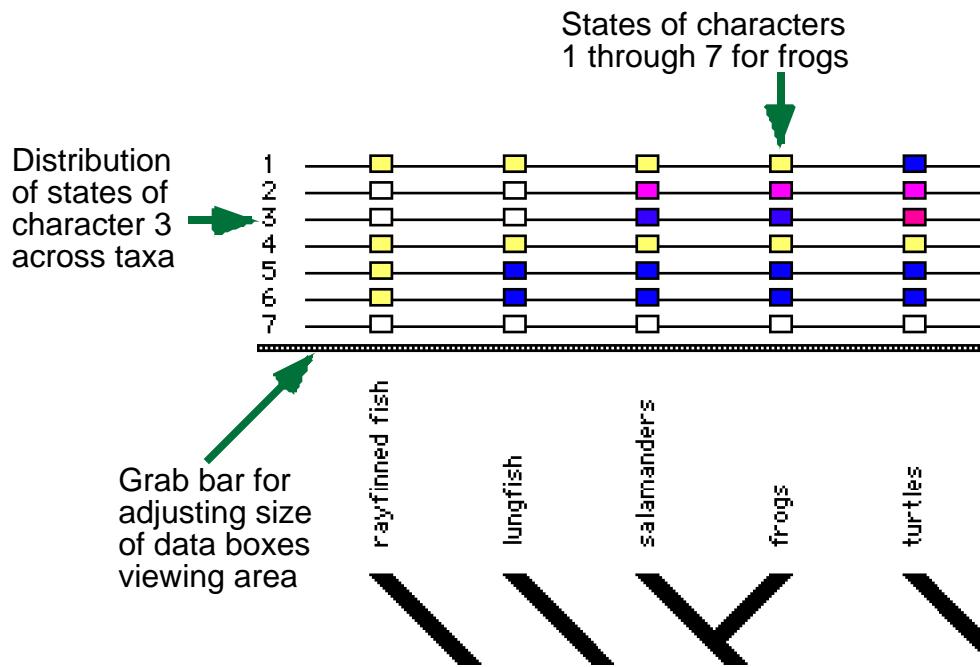
You can, using the Trace Labeling dialog box in the Display menu, ask MacClade to write textual descriptions of the states at each branch in addition to the patterns and colors. If you wish, these textual descriptions can use full state names (as shown in the figure, below) or state symbols.



A traced character labeled with full state names

Character Data Boxes

You can get a graphical overview of the data matrix by selecting **Data Boxes** from the **Show** submenu of the **Display** menu; reselect the **Data Boxes** menu item to remove the boxes. The Data Boxes look something like this:



You can scroll through the Data Boxes using the scroll bar on the right.

In order to get an overview of more than the default number of characters, you can expand the Data Boxes area. Do this by placing the cursor over the grab bar at the bottom of the Data Boxes; when the cursor changes to , drag the bar down. You can also shrink the Data Boxes section by dragging the bar up.

Data Boxes can be printed from the **Print Other** submenu (in which case the order of taxa will match that in the data matrix) or the **Print Tree** dialog box (in which case the order of taxa will match that in the current tree). See [Chapter 25](#) for details.

Summary of all changes on branches: Trace All Changes

MacClade has two methods for making a summary over all characters of the reconstructed changes in the tree: the **Trace All Changes** display on the tree, and the charting of **State Changes & Stasis** in the **Chart** menu. State Changes & Stasis shows the number of state *i* to state *j* changes in a table or bubble chart to give an indication of the relative frequency of various reconstructed changes, and is described in [Chapter 20](#).

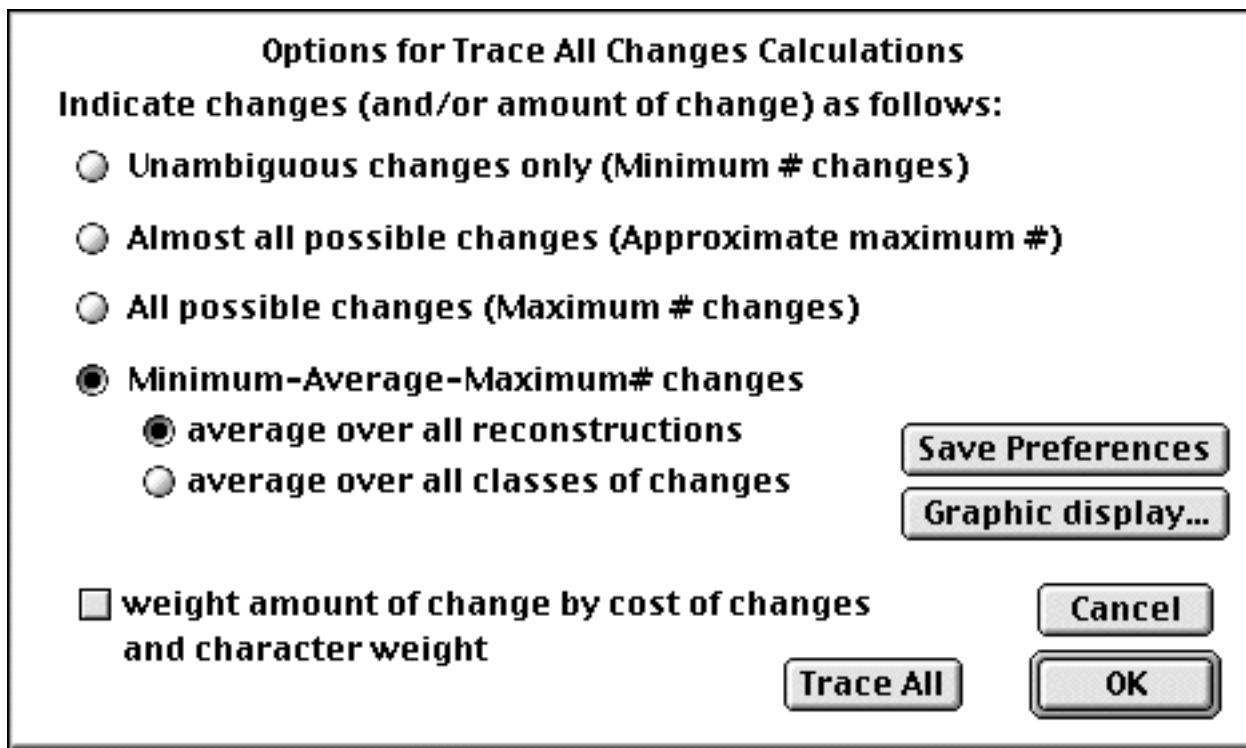
To activate **Trace All Changes**, choose that menu item from the **Trace** menu. Trace All Changes shows the number of characters changing on each branch or the amount of the tree's length on each branch (depending on whether changes are weighted or not). It can also show which characters are reconstructed as changing along each branch. It displays the results by shading the branches of the tree, listing the branches' lengths, or putting bars across them.

For a discussion of the relationship between the number of changes reconstructed on a branch and the evidential support for the clade above the branch, see [page 369](#).

Stratigraphic characters are excluded from the calculations of Trace All Changes.

Options for Trace All Changes calculations

The **All Changes Options** dialog box (in the **Trace** menu) allows you to select options in the calculations of Trace All Changes. You must make two decisions: (1) how ambiguity in changes is to be dealt with (main four radio buttons in the upper part of dialog box), and (2) whether the amount of change is to be weighted or not (choice at bottom).



Dealing with ambiguity

If several states may be placed with equal parsimony at a node, then whether a change is placed on the branch between this node and its ancestor may depend upon which of the alternative states is assigned to the node (see [Chapter 4](#)). Because ambiguity in the reconstruction yields ambiguity in changes, the **All Changes Options** dialog box gives four choices:

1. Show unambiguous changes only. An unambiguous change is counted as occurring on a branch if the sets of states reconstructed at a node and at its ancestor do not overlap at all, because then there must be a change on the intervening branch regardless of which parsimonious states are assigned to these nodes. (Note: This option indicates a change whenever it is unambiguous that some change occurs, but it may be ambiguous as to exactly what change occurs; the "unambiguous changes" option of the State Changes & Stasis chart is different, because that indicates a change only when it is unambiguous as to exactly what change occurs.)
2. Show almost all possible changes (most ambiguous and unambiguous). This option counts a change if the state sets of a node and its ancestor are not identical. If the state sets are not identical, then there exists at least one parsimonious reconstruction in which a change occurs on the branch. This criterion will detect most but not all instances in which parsimony allows a change on the branch. Even when descendant and ancestor state sets are identical, there may exist a reconstruction placing a change on the branch between them. To discover these cases you need to use the third option.
3. Show the maximum amount of change allowed by parsimony. This choice finds all instances in which parsimony allows a change on the branch. If there exists a parsimonious reconstruction that places a change on the branch, MacClade will count a change.

4. Show minimum, average, and maximum amount of change allowed by parsimony. This option considers both unambiguous changes and all possible changes to calculate the minimum and maximum amounts, and at the same time calculates the average amount of change as measured over the various possible reconstructions. There are two options regarding averages: average over all reconstructions, and average over all classes of changes. These two options are discussed in [Chapter 4](#). How the minimum, average, and maximum are displayed is discussed below under "[Branch display](#)", below.

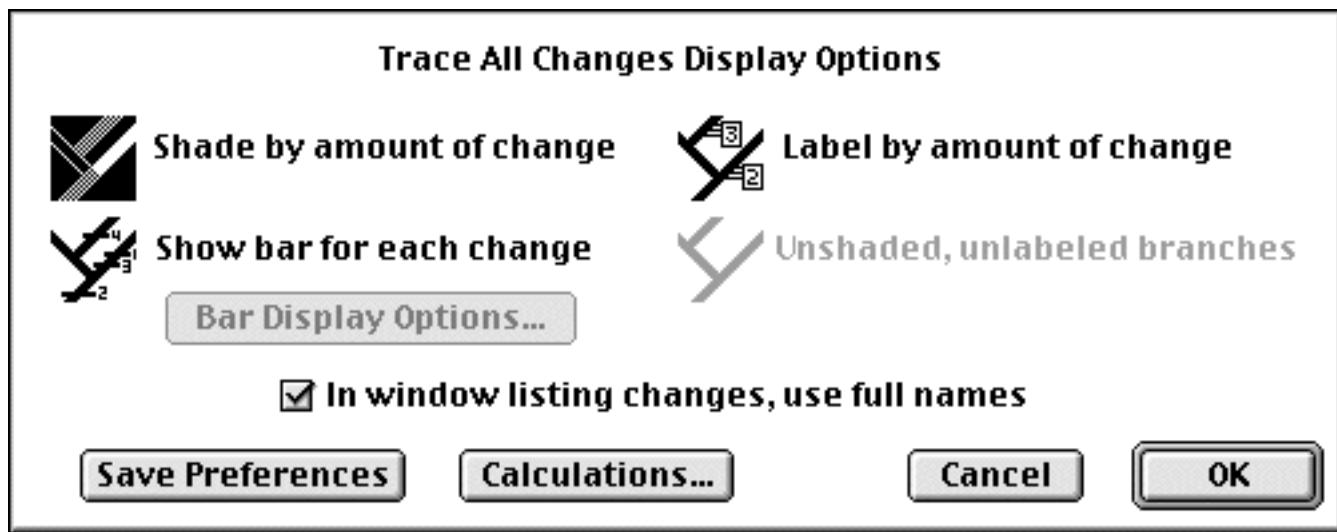
More information on how these calculations are done can be found in [Chapter 4](#). The third and fourth options cannot be used if there are polytomies or other factors that prohibit explicit or implicit examination of all MPRs.

Weighted changes

By default, the total amount of change indicated on a branch by Trace All Changes is unweighted, and is simply the number of changes on the branch. One change is counted for each character that shows some change, from the node at the beginning of the branch, to the node at the end of the branch, regardless of the type of change or the character's weight. In contrast, if "weight amount of change..." in the **All Changes Options** dialog box is checked, then the amount is weighted by both the character's weight and by the cost of the change according to the character's transformation type. Thus, a change from state 0 to 3 in an ordered character of weight 5 would add 15 to the amount of change on the branch. When the unambiguous changes option is selected, the minimum weighted amount of change is indicated. Weighting is not allowed when the second option above is selected ("Almost all possible changes").

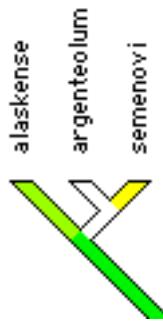
Branch display

MacClade can indicate the amount of reconstructed change on a branch by drawing branch lengths proportional to the amount of change, or by giving an indication on the branch through coloring or labeling of changes. The manner in which MacClade displays the branches can be controlled by the **Trace Labeling** dialog box in the **Display** menu:



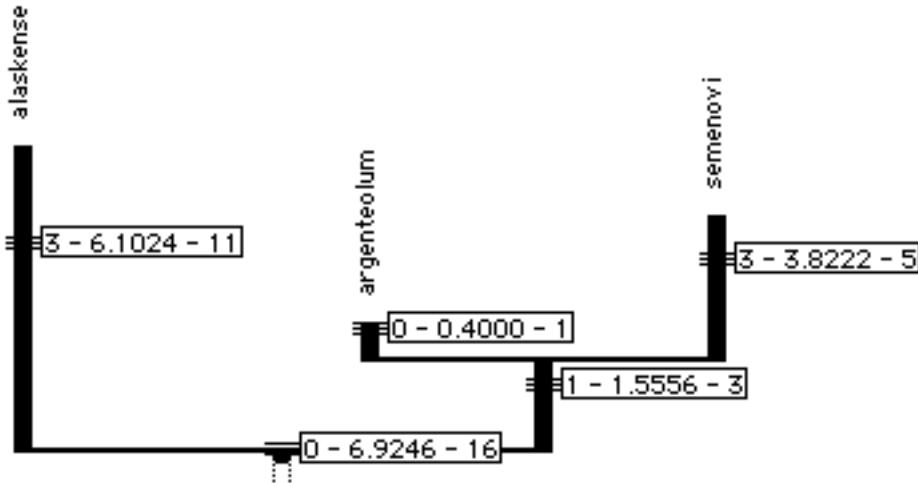
There are four ways that MacClade can display branches in Trace All Changes mode. (There are some additional display modes that are available only with printed trees; see ["Printing trees" on page 453](#).)

First, the amount of change can be indicated by shading the branches, with the pattern or color indicating the number of changes (the option "Shade by amount of change"):



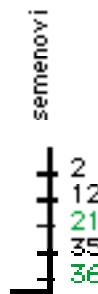
You can change the patterns and colors used to shade the tree using the same techniques as for changing the patterns and colors used to trace character evolution, as described above (["Changing patterns or shades or colors" on page 355](#)). When the minimum-average-maximum option is selected in the All Changes Options dialog box, the shading indicates the average amount of change.

Second, the amount of change can be indicated directly by a numbered box (the option "Label by amount of change"). If minimum, average, and maximum changes are being shown, then all three of these will be written into the box:



If you choose to label branches by the amount of change, and the changes on a particular branch are not calculated (for example, because the branch is part of a polytomy), then the branch will be labeled "NC" rather than "not calc.". This was done to prevent overlapping labels on the screen.

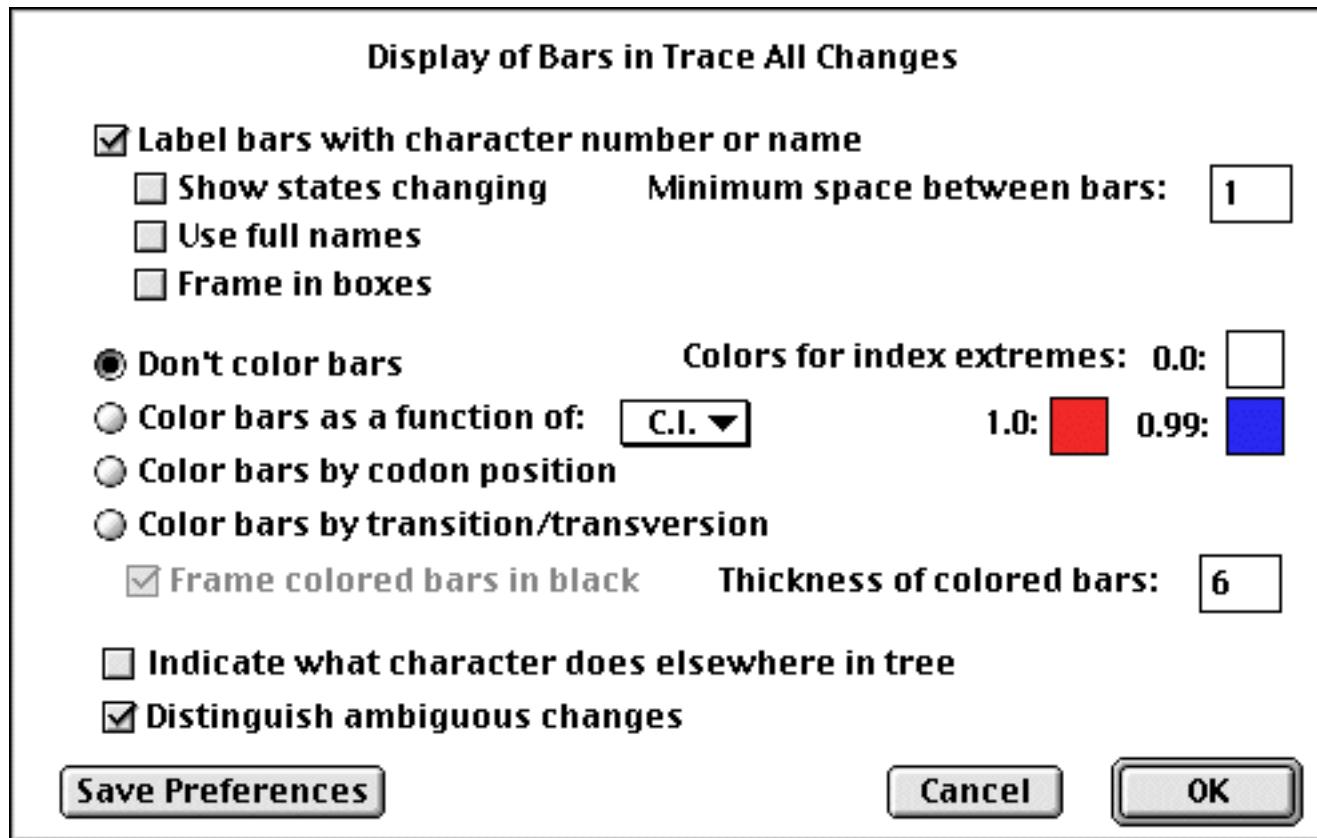
Third, the individual changes can be displayed directly on the tree by placing bars across the branches, one bar for each change, with the number of the character changing listed beside the bar (the option "Show bar for each change"):



If you use bars, you may wish to adjust the shape and size of the tree, using the **Tree Shape & Size** dialog box in the **Display** menu (["Branch lengths proportional to number of reconstructed changes" on page 338](#)), so that the markings on the branches are easier to see. You can request that MacClade draw the length of each branch proportional to the number of changes reconstructed along the branch.

Fourth, you may request that the branches remain black and unlabeled. This option is available only from the **Print Tree** dialog box, and is intended for trees that are drawn with branch lengths proportional to the number of changes.

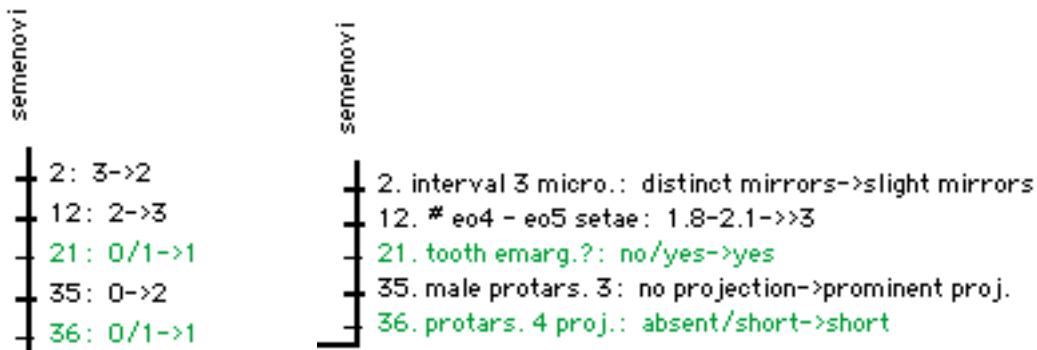
If you do show a bar for each change, there are many options as to how the bars are drawn, available in the **Bar Display Options** dialog box under the **Display** menu:



If you display the characters changing along a branch with bars, you can choose to have the bars unla-

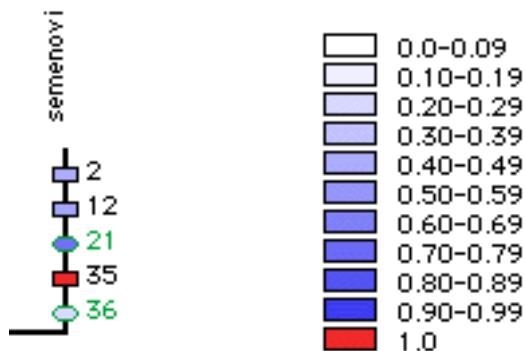
beled.

The states that change can be indicated, and full character and state names can be listed:



In this mode, however, labels from adjacent branches will often overlap.

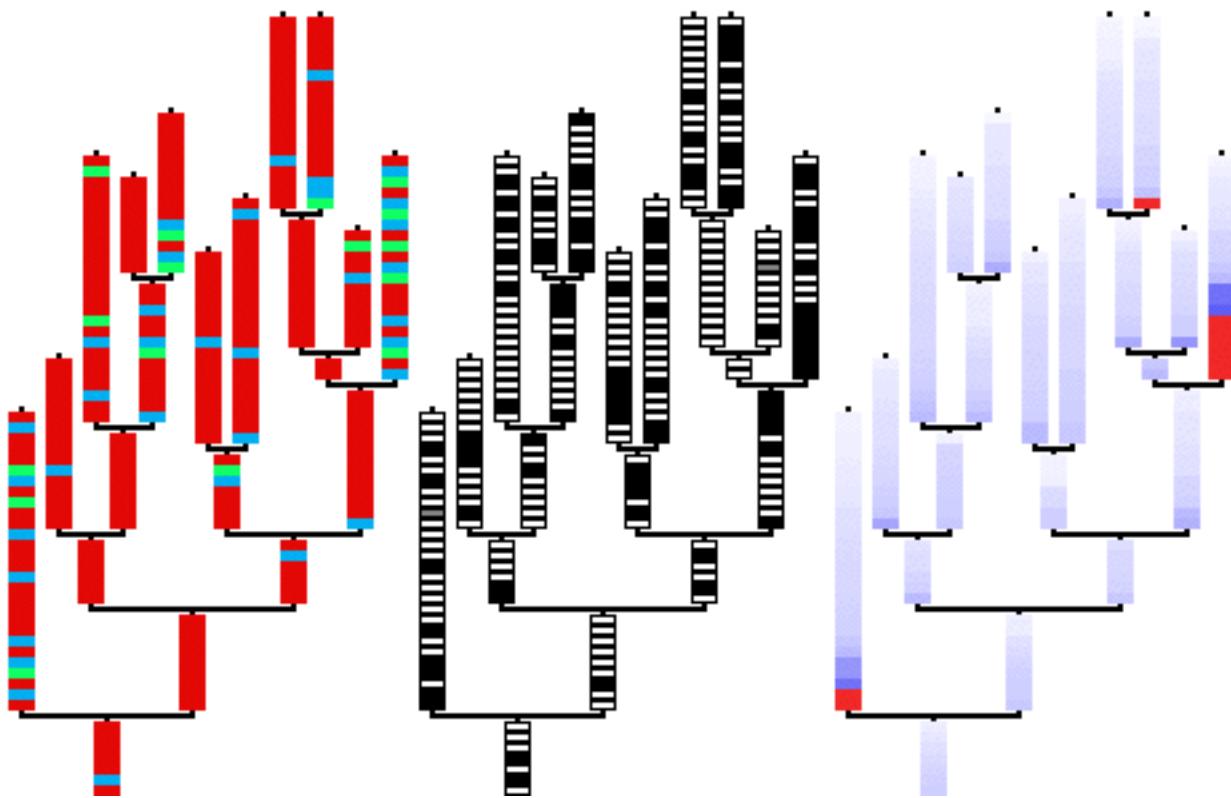
By default, bars are no more than thin lines. You can, however, ask MacClade to thicken and color them according to various criteria. For example, you can ask MacClade to shade the bars as a function of the consistency index of the character, with white being the poorest value, red being the best, and dark blue being next-to-best (as shown on the right):



In this mode, characters with ambiguous changes are represented by ovals and green text. The colors used can be altered by touching on the "Colors for index extremes" boxes in the dialog box.

If the data are nucleotides and protein coding, the bars can be colored green for first positions, blue for seconds, and red for thirds. Nucleotide data can also have the bars colored depending upon whether the reconstructed change is a transversion or not.

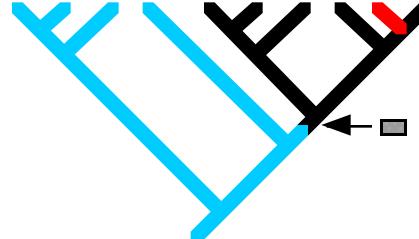
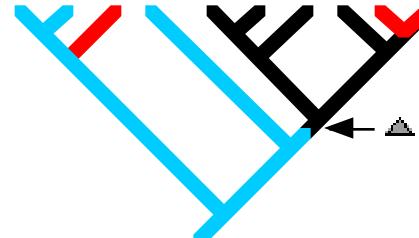
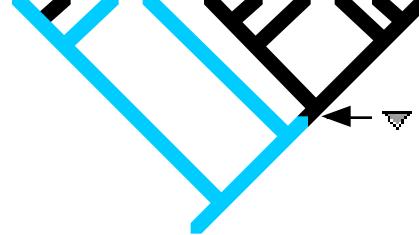
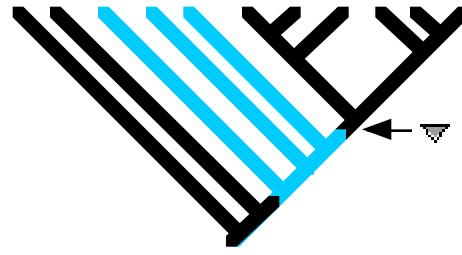
The following three images represent three different colorings of bars for one nucleotide data matrix:

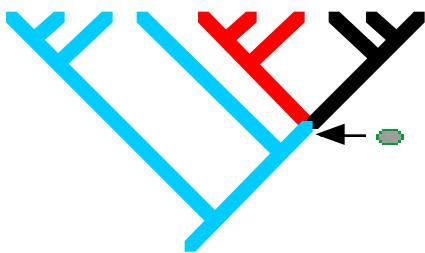


In each case, the bars were asked to be unlabeled, with no space between them. The left figure shows the changes colored by codon position, the middle figure shows transversions in black, and the right figure shows characters colored by consistency index, with the characters sorted by CI using the sort tool in the list window ([page 177](#)).

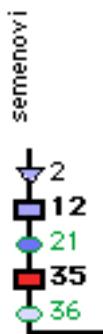
If you check "indicate what character does elsewhere on tree", then MacClade will indicate something about that character's reconstructed evolution elsewhere on the tree. The symbols used and their meaning are shown in the following table:

| Symbol | Meaning | Examples |
|--------|---|----------|
| ■ | Derived character state is unique in the entire tree, and is unchanged above this branch. | |

| Symbol | Meaning | Examples |
|--------|---|--|
| ■ | Derived state at node is unchanged throughout the rest of the tree outside the clade, but within the clade the character changes to another state (with no reversals to ancestral state). |  |
| ▲ | Homoplastic change above: Within clade a different state arises that is also found outside the clade, as either a reversal to an ancestral state or a convergence to a state found outside the clade. |  |
| ▼ | Homoplastic change below: Derived state is also found outside this clade. It thus represents either a reversal to an ancestral state, or this state is convergent on a state found outside the clade. |  |
| | |  |

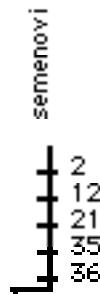
| Symbol | Meaning | Examples |
|--------|---|--|
| ◆ | Homoplastic change both above and below. |  |
| ● | Ambiguous change. Not present in some MPRs. |  |
| ■ | Derived state unclear: In all MPRs a change occurs along branch, but the derived state varies among MPRs. | |

For example, on the following branch:



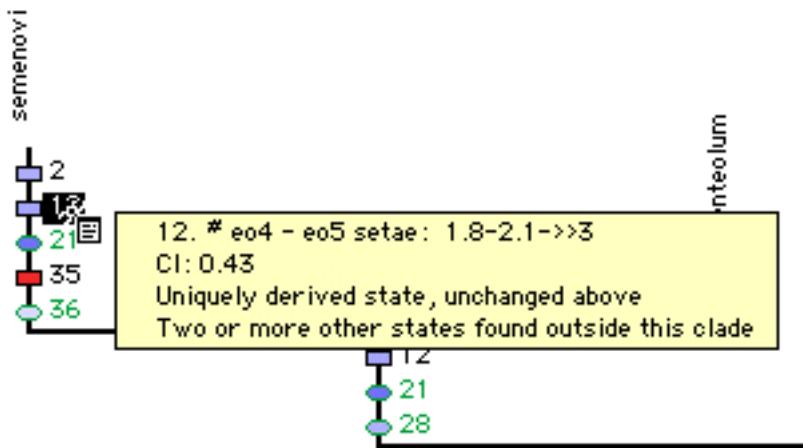
the derived states for characters 12 and 35 represent a uniquely derived, nonhomoplastic state.

By default, any ambiguous change ([page 359](#)) is indicated in green, but you can ask MacClade not to distinguish ambiguous changes in that way, if you turn off "Distinguish ambiguous changes":

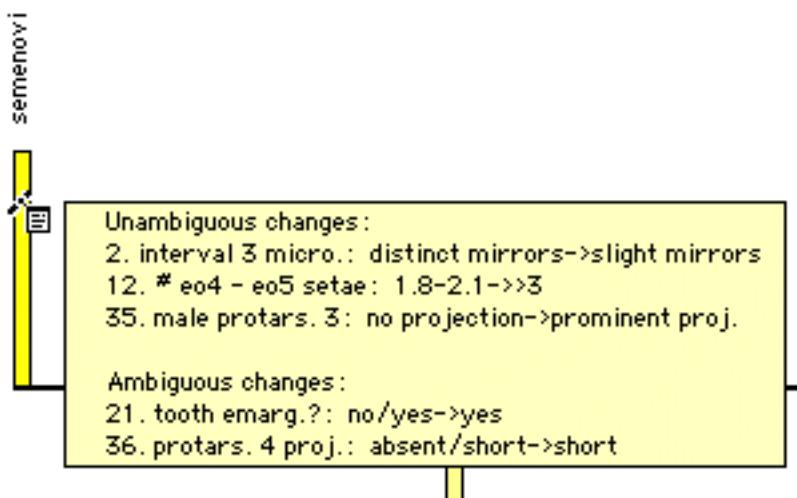


More information about the changes occurring on branch

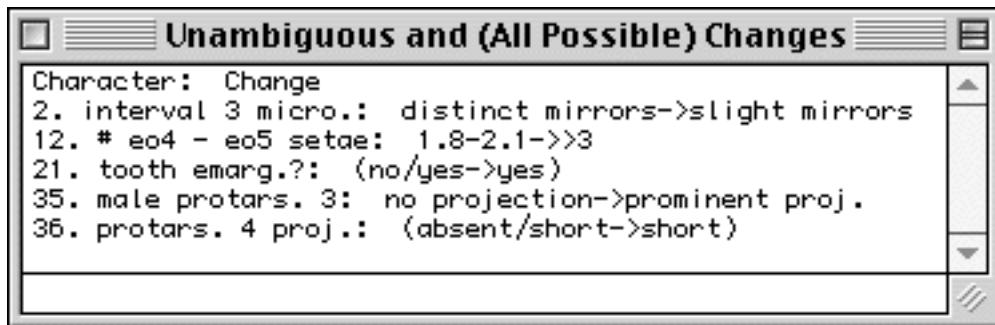
Whenever bars are shown in Trace All Changes mode, you can get detailed information about a character by touching the branch list tool on the character number:



If no bars are shown, this tool instead lists all of the changes occurring along that branch, with some information:



Whether or not bars are shown, if you switch the mode of the tool to **Separate Window** mode (available in the tool's pop-up menu), then when the tool is touched on a branch, a small, scrollable summary window will then appear, listing the changes that occur along that branch.

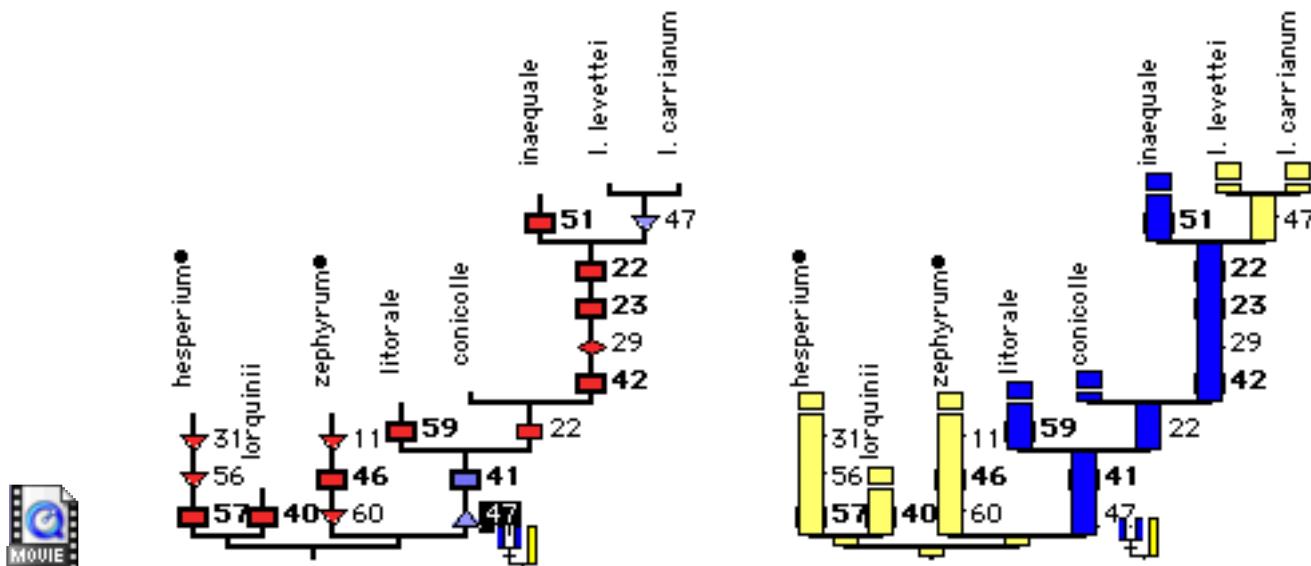


Using the **Trace Labeling** dialog box in the **Display** menu, you can control whether full state names or state symbols are used in this window. If you are using one of the options to trace almost all possible, all possible, or minimum-average-maximum, and have asked MacClade to distinguish ambiguous changes, all ambiguous changes (i.e., all those other than unambiguous changes) will be shown in parentheses.

A listing of the changes on all branches can be printed or saved to a text file using **Branch Lists** in the **Print Other** or **Save Text File** submenus. See [Chapter 25](#) for details.

Tracing a character shown in Trace All Changes

If you wish to temporarily show a character that is represented by a bar in Trace All Changes, then use the Temporary Trace tool and touch on the character's number, as shown below.



If you switch the mode of this tool by unchecking **Temporary Trace Mode** in the tool's pop-up menu, then when you use the tool on a character's number, Trace All Changes will be turned off and that character will be traced.

If you double-click on a line in the list window, Trace All Changes will be turned off and the corresponding character will be traced on the tree.

Is the number of changes indicative of support for a clade?

Even though the changes along a branch listed under the unambiguous changes option can often be considered as apomorphies that unambiguously support the clade above the branch, this is not always a good interpretation. For instance, if the character subsequently reverses in one taxon within the clade, it would be inappropriate to claim that the character supports the monophyly of the clade, except in the context of other characters that place the reversing taxon within the clade. Likewise, a tree with many unambiguous changes is not necessarily a good tree, for some of the unequivocal character state changes that occur on the tree might be avoided by a shorter tree. Thus whether or not a character changing on a branch gives support to the clade above the branch may depend upon the context of other characters.

If you want to assess the support for a clade, it is more reasonable to compare the length of the shortest trees with that clade and without that clade (Bremer, 1988; Miyamoto and Boyle, 1989; Hillis and Dixon, 1989; Donoghue et al., 1992), or to use bootstrapping (Felsenstein, 1985c; Sanderson, 1989; Swofford and Olsen, 1990).

Problems with soft polytomies

When a tree has soft polytomies, Trace All Changes does not attempt to calculate changes occurring on the branches of a polytomy. Because the changes that would be calculated on these branches depend upon how the soft polytomy might be resolved, it is difficult to decide where to place changes upon them (["Locating and summarizing changes of character state" on page 103](#)). These branches are shaded with an equivocal pattern, and a warning is given to the user that MacClade did not attempt to calculate changes on these branches.

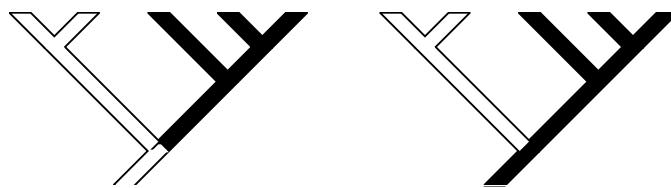
Problems near the root

The root of the whole tree will never have changes indicated on it. A change on the root branch could be discovered only by assuming something about states in ancestral lineages below the root, or about states in related groups. MacClade works with the states observed in taxa that are included in the data matrix, and does not assume anything about more distant relatives or ancestral states. MacClade, for instance, does not use ANCSTATES as does PAUP*. MacClade assumes nothing about what is below the root, and thus will not indicate changes on the root branch.

The right and left branches descendant from the root cannot have unambiguous changes except in characters that are irreversible (or have step matrices that are asymmetrical or violate the triangle inequality). In most characters, if the two nodes descendant from the root node are reconstructed as having different states, then the root will be equivocal and it will be ambiguous as to whether the change occurred on the left or right side. For instance, if the right descendant branch of the root is reconstructed to have state 1, whereas the left branch has state 0:



then there is ambiguity as to whether the root had 0 changing to 1 up the right side or the root had 1 changing to 0 up the left side:



On the other hand, if the two nodes descendant from the root have the same state, then the root will share this state and no changes are required.

Even when no changes are indicated by Trace All Changes as occurring on the two branches descendant from the root, you might be interested in knowing how these two branches differ. The branch-information tool, when applied to either of these two branches, will list those characters in which there is an unambiguous or ambiguous difference between these branches.

Branch lengths

There is currently one statistic calculated by MacClade that might be considered a measure of branch length ([page 50](#)): the amount of change reconstructed on a branch by Trace All Changes. (MacClade does not yet allow users to specify their own branch lengths, except in the special context of Evolve Characters, [page 435](#).) It is important that you understand the ambiguity and problems involved in interpreting this reconstructed amount as a branch length.

First, there may be several equally parsimonious reconstructions of character evolution, some of which place a change on a branch, others of which don't. The options in Trace All Changes can be used to indicate the minimum and maximum amount of change allowed by parsimony on a branch.

Second, MacClade does not indicate any length on certain branches because of lack of information or calculation difficulties. Changes on the two branches coming off of the root of the tree are often shown with zero branch length, even when the two clades differ in the characters. This is a simple consequence of the ambiguity regarding where changes occur, and is discussed under "Problems near the root", above. Zero branch length is also always indicated on any branch arising from a soft polytomy (discussed under "Problems with soft polytomies", above). Although many users may be disappointed with this, they should realize there is no simple solution to this problem. See [Chapter 4](#) for more explanation of the difficulties in reconstructing changes with soft polytomies.

Third, the true amount of change that actually occurred on a branch may be different from that reconstructed. See [Chapter 3](#) and [Chapter 4](#) regarding error and confidence.

Fourth, the true amount of change may not itself be an accurate estimate of the length of time a branch existed.

Display and printing of branch lengths

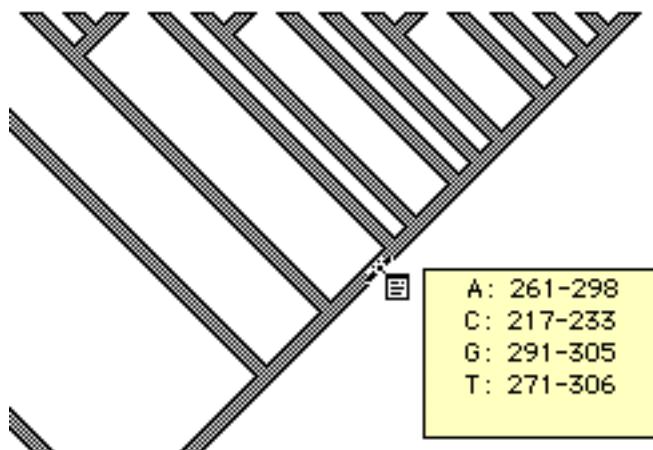
The amount of change reconstructed on each branch can be displayed in the tree window as discussed in the preceding sections, and the length of branches can be drawn proportional to number of changes, as described in "[Tree display commands](#)" on page 334. There is also a special printing feature that prints trees with their branches having lengths proportional to the amount of reconstructed change. See "[Printing trees](#)" on page 453 for a description of this feature.

Summary of all states along each branch

Choosing **Trace All States** from the **Trace** menu will cause MacClade to reconstruct ancestral states of all included characters. The results are visible by clicking on a branch of interest using the branch-list tool



(); a box will appear listing the state information:



By default, the frequency of reconstructed states will be presented.

If you switch the mode of the tool to present the information in a separate window (this mode is available in the tool's pop-up menu), then you will have two options for the data presented, both available under the tool's pop-up menu. **State Frequencies** will display the frequency of each state (0, 1, 2, or A, C, G, T, and so on) over all included characters at that branch; this is perhaps most useful for molecular data in which states are comparable between characters. When the reconstruction is equivocal for a character at a branch, then the frequency listing reflects this by showing the minimum and maximum frequencies allowed by the ambiguity. For stratigraphic characters, MacClade treats the state as equivocal whenever multiple states are placed at a node, even if they are placed there as a result of an observed polymorphic taxon fixed in ancestral position. **States in Each Character** will list, for each included character, the reconstructed state at that node.

A character with all missing data in the observed taxa in the current tree will be reconstructed as having any possible state at each node in the tree for molecular data (e.g., the MPR set at each node will be "A or C or G or T"). Thus these MPR sets will contribute to states and frequencies listed by Trace All States.

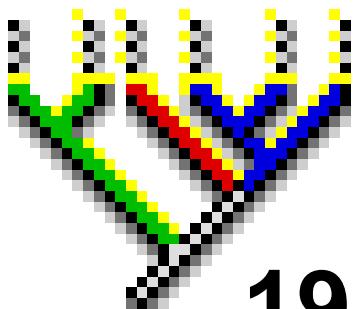
Confirming the validity of the parsimony algorithms

In programming MacClade, the proper functioning of the discrete-character parsimony algorithms was tested as follows.

- The source code was thoroughly checked and rechecked.
- Both ancestral state reconstructions and counts of treelength were examined by eye in a number of test data files, and also cross-checked against results from MacClade 2.1 and PAUP 3.0.
- In test versions and in the released version of MacClade 4, traced characters are checked to see that none of the final sets are empty, nor that they contain illegal values that should be cleared in the final sets.

Faulty algorithms could express their errors by generating empty sets or adding illegal values.

- In test versions, a thorough consistency check was used that moved through all characters, comparing the calculated number of steps in the character with the number of steps recalculated as follows. If the tree had no polytomies nor observed taxa designated as ancestors, then explicit examination of each MPR was used to make all most-parsimonious reconstructions, in each counting the number of steps and comparing to the number calculated with MacClade's usual step counting algorithms. This consistency check not only detects faulty reconstructions of character evolution, but also checks the routines to explicitly examine all MPRs. If the tree had polytomies or observed ancestors, then the states at nodes were fixed successively from the root toward the tips to generate various reconstructions of ancestral states. First, the lowest-valued states allowed were fixed, and the number of steps in the resulting reconstruction compared with that previously calculated. Second, the highest-valued states were fixed, and the number of steps compared with the previous. This consistency check can detect faulty reconstructions, and also tests the algorithms dealing with the fixing of states at nodes (paintbrush tool) and how they interact with polytomies and observed ancestors.
- In test versions of MacClade 3, a consistency check was incorporated that compared MacClade's two separate sections of code used in reconstructing characters whenever an unordered, ordered, or irreversible character was traced.
- After the assembly language sections of MacClade 3 were translated to C for version 4, results from a variety of characters were checked against those from version 3.



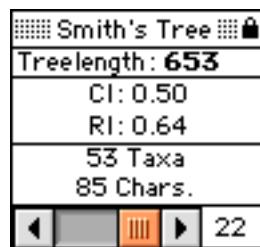
BASIC TREE AND CHARACTER STATISTICS

19

Display of statistics

The most parsimonious reconstructions of ancestral states imply a minimum number of steps in each character for the current tree. This minimum number of steps is one of the basic statistics used by many phylogenetic systematists as a measure of the fit between the character and the tree. By standardizing this number of steps in various ways, various indices like the consistency index can be obtained. By summing the number of steps over all characters, perhaps using weights as well, the **treelength** is obtained.

Those statistics that apply to the current tree as a whole (e.g., treelength, ensemble consistency index, and retention index) are displayed in the tree legend.



Tree legend displaying treelength, CI, and RI

Those statistics that apply to each character can be displayed in various charts ([Chapter 20](#)) and seen in the character list window (available from the **Characters** menu). In the following figure, note the columns for steps, CI, and RI.

| Character | Type | Weight | States | Steps | CI | RI |
|--------------------------|---------|--------|--------|-------|------|------|
| 1 silver spots | ordered | 1 | 2 | 1 | 1.00 | 1.00 |
| 2 interval 3 micro. | ordered | 1 | 4 | 10 | 0.40 | 0.89 |
| 3 interval 4+5 micro. | ordered | 1 | 4 | 20 | 0.70 | 0.87 |
| 4 interval 6+7 micro. | ordered | 1 | 4 | 14 | 0.79 | 0.86 |
| 5 outer mirror | ordered | 1 | 2 | 3 | 1.00 | 0.0 |
| 6 silver spot | ordered | 1 | 2 | 8 | 1.00 | 1.00 |
| 7 elytral microsculpture | ordered | 1 | 3 | 20 | 0.40 | 0.54 |
| 8 suborbital setae | ordered | 1 | 3 | 27 | 0.56 | 0.64 |

Number of steps

The number of steps required for each character is shown in the legend of the character traced, as described in [Chapter 18](#), and in the character list window. The number of steps in a character is the summed cost of all changes in a most parsimonious reconstruction ([Chapter 3](#)). For character i , the number of steps, s_i , can be calculated by the following formula.

$$s_i = \sum_{j,k} c_{jki} n_{jki}$$

where c_{jki} is the cost of a change from states j to k in character i , and n_{jki} is the number of j to k changes on the tree in a most parsimonious reconstruction of ancestral states for character i . If there are multiple equally parsimonious reconstructions, then any one could be used for the formula above, for if they are equally parsimonious they must give the same sum. (Compare the number of steps to number of changes, discussed below.) See [Chapter 4](#) for descriptions of the algorithms used to count steps.

NOTE: "Steps" may not be an entirely appropriate term to use, because the number of steps as defined above could be a number like 1.6 (two changes each with costs 0.8 according to the character's step-matrix transformation type), in apparent contradiction to the implication of "step" as a discrete change. The term "step" is a remnant from older computer programs in which it was more appropriate.

The number of steps in each character is shown both in the character list window when treelength counting is turned on and in the Trace Character legend.

When terminal taxa are polymorphic, MacClade counts implicit changes within these terminal taxa. Thus for instance, suppose a terminal taxon has states 2 and 5 in an ordered character and MacClade reconstructs state 1 to be at the branch immediately ancestral to the terminal branch. MacClade would then count 1 step from state 1 to 2 along the terminal branch, then 3 steps from state 2 to 5 within the terminal taxon itself. Because MacClade does not have algorithms directly allowing ancestral polymorphisms in branches below the terminal taxa, the immediate ancestor is presumed to have a single state, and thus the diversity of states within the terminal taxon is evolved within the terminal taxon (see [Chapter 4](#)). MacClade assumes the most parsimonious path of evolution possible within the terminal taxon. Because it is difficult to determine the most parsimonious path when the taxon has three or more states in a character of user-defined type, the number of steps counted may be underestimated. The reason is that MacClade uses only the smallest distance between two observed states as an indication of the length within the taxon (as discussed in [Chapter 4](#)). In this case MacClade gives an error message, and puts an asterisk (*) beside the number of steps in the character list window to warn the user.

Treelength

The total treelength is calculated as the sum of the number of steps for the individual characters (s_i) multiplied by their respective weights (w_i):

$$\text{Treelength} = \sum_{i=1}^n w_i s_i$$

(In this formula, n is the number of characters.) Treelength includes steps that must have occurred within polymorphic terminal taxa, except occasionally for characters of a user-defined type. For characters of user-defined types, changes within terminal taxa polymorphic for more than two states may not be counted accurately (see [Chapter 4](#)); in this case the treelength will be underestimated and MacClade will put a "+" in the treelength box to indicate that the actual treelength is higher than the number given. Simi-

larly, treelength will be marked by a "+" if the tree contains polytomies and the soft polytomies option is chosen, for under this option the treelength listed will likely be an underestimate of the minimum possible treelength for any fully dichotomous resolution of the tree ([Chapter 4](#)).

To turn treelength counting off or on, choose **Treelength** in the Σ menu. Turning off the count is useful with large data sets when you want to make a lot of branch rearrangements and don't need to know treelength until you're done. Having the tree counted between each branch move can slow things down, especially if there are user-defined or Dollo characters.

WARNING: Users moving between PAUP* and MacClade may notice situations in which PAUP* gives a different treelength. See "[Treelength](#)" on page 476 for details.

Polytomies and treelengths

Uncertain-resolution ("soft") polytomies pose particularly difficult problems for calculating treelength as noted by W. Maddison (1989). See [Chapter 4](#) for a discussion of the difficulties of calculating treelength with soft polytomies.

Number of changes

The number of changes required for a character differs from its number of steps in that the former is not weighted by the costs of the reconstructed changes. The number of changes for different characters can be displayed in the Character Steps/etc. chart, or summed for all characters and shown as the Tree Changes (see below). The number of changes in each character is shown in the character list window when tree changes counting is turned on.

In a most parsimonious reconstruction of character evolution, a change is required whenever a node has a different state from its immediate ancestor. The number of changes in a character on the tree is counted by MacClade as the number of instances in which a change is reconstructed as occurring on a branch. Implicit multiple changes within a branch are not counted separately. As noted in [Chapter 4](#), if there is ambiguity in the tracing of character evolution, such that an equivocal assignment is given to some branches, then the alternative reconstruction of character evolution may have different numbers of changes ("[Changes](#)" on [page 102](#)). Therefore, MacClade may sometimes indicate a range of changes, for instance "5-7 changes". Changes within polymorphic terminal taxa are *not* counted (recall that their costs are counted into the number of steps). MacClade cannot calculate the number of changes with polytymous trees or trees that have observed taxa fixed as ancestors.

Tree changes

A statistic related to treelength is the number of changes in the tree. It is the unweighted sum of the number of changes in all the characters. The number of changes differs from the treelength in that it is not weighted either by the cost of each change, nor by the weights of the characters. It can be requested by checking the **Tree Changes** item in the Σ menu. The results are presented in the tree legend. The calculations, which can be time consuming, can be canceled by Command-period (⌘-.).

As noted above, the number of changes in a character may differ with alternative reconstructions of character evolution. MacClade sums all the minima, and all the maxima, for each character, and displays a minimum and maximum total value for tree changes (e.g., "95–102 changes"). Changes within polymorphic terminal taxa are *not* counted. Recall that their steps are counted in the treelength. If there are polymorphic terminal taxa, then MacClade puts an asterisk ("*") beside the number of changes to indicate that changes were not counted within terminal polymorphisms. Because calculation of tree changes uses explicit or implicit examination of all MPRs, MacClade cannot calculate the tree changes for polytymous

trees or trees that have observed taxa fixed as ancestors.

One context in which tree changes may be useful is to deal with missing characters, as suggested by W. Maddison (1993). If you had a character with states tail red, tail blue, and tail absent, you might want to reconstruct character evolution first considering gains and losses of tails, then secondarily considering changes in tail color within those parts of the tree reconstructed as having tails. To do this, a step matrix could be created that counted one step for a change in tail color from red to blue or vice versa, but then 100 steps for a gain or loss of a tail (change from absent to red or blue, or vice versa). This would reconstruct character evolution as desired, but would count 100 steps for the gain or loss of a tail. If you wanted to give tail gain/loss primacy over tail color, but wanted all the changes to count equally, you could use tree changes instead of treelength (see W. Maddison, 1993).

Minimum and maximum possible steps

You can ask MacClade to show the minimum and maximum conceivable number of steps in each character and the sum over all characters, by choosing the **Minimum Possible** or **Maximum Possible** items in the Σ menu. Values for the sum over all characters are displayed in the tree legend; those for individual characters are displayed in the character list window (accessible in the **Characters** menu).

The minimum conceivable number of steps in a character is the smallest number of steps that could be obtained in the character were the best tree found for that character by rearranging the terminal taxa. Thus if four taxa have states 0, 0, 1, and 1 for an unordered or ordered character, the fewest number of steps would be 1, by grouping the two taxa with 1s together. In calculating the minimum conceivable number of steps, polymorphic terminal taxa are not allowed to split up to avoid steps within the terminal taxa. That is, if one taxon has states 0 and 1, and the remaining three taxa have states 0, 1, and 1, then the smallest number of steps would be 2, because one step is required within the polymorphic terminal taxon and one in the remainder of the tree. Minimum number of steps are not calculated for characters of stratigraphic, Dollo, or user-defined types.

The number reported in the tree legend for the total minimum number of steps is the sum over all characters. This does *not* mean that there exists a tree with this number of steps that could be obtained by rearranging the branches, for the rearrangement required to yield the minimum for one character might not be the same as the minimum required for another character.

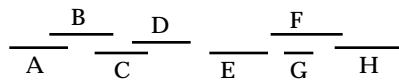
The maximum conceivable number of steps is the largest number of steps that could be obtained in the character were the worst tree found for that character by rearranging the terminal taxa. Thus, if four taxa have states 0, 0, 1, and 1 for an unordered or ordered character, the largest number of steps would be 2. Polymorphic terminal taxa are held intact, so that if one taxon has states 0 and 1, and the remaining five taxa have states 0, 0, 0, 1, and 1, then the largest number of steps would be 3. Maximum number of steps are not calculated for characters of irreversible, stratigraphic, Dollo, and user-defined types. As with the total minimum, the total maximum number of steps reported in the tree legend does not imply that there exists a tree with that number of steps.

Calculation of minimum and maximum steps

For unordered and ordered characters, MacClade uses its "soft" polytomy calculations ([Chapter 4](#)) to obtain the minimum number of steps conceivable for a character, by measuring the number of steps in the character in a tree that is an entirely unresolved bush. Irreversible characters are done using the soft polytomy calculations for ordered characters, because the smallest maximum to the largest minimum among the terminal node downpass sets is the minimum conceivable length for irreversible characters as well.

The maximum number of steps was defined above as the number of steps on the tree requiring the most steps for the character. A hard polytomous bush is the least parsimonious tree conceivable. On that basis,

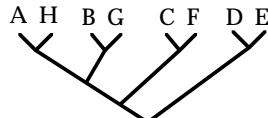
one could argue that it is appropriate to use the number of steps on a hard polytomous bush as the maximum conceivable number of steps for a character. In fact, MacClade uses the hard polytomy calculations to obtain the maximum number of steps conceivable for unordered and ordered characters. However, one might prefer to use as a standard the least parsimonious *dichotomous* tree for the character. The distinction may be unimportant, for it is apparently the case that the least-parsimonious dichotomous tree for a given character has the same number of steps as a hard polytomous bush. This can be proven for ordered characters, as follows. Suppose we have 9 taxa named A through H, with observed state ranges diagrammed as follows (lower values to left, higher to right):



A least parsimonious dichotomous tree can be constructed that places the two terminals with most distant state sets as sister species, and then pairing the most distant among the remaining terminals, and so on, until none or one terminal remains. We now have the taxa paired:



These pairs of taxa can be joined together into a full tree in any way:



As one does a downpass ([Chapter 4](#)) on this tree for this character, all of the steps in the tree will be counted within these sister pairs, as (for each pair) the state set of one terminal taxon of the pair is combined with the state set of the other terminal taxon to yield the state set at their immediate common ancestor. The number of steps so counted will be the exact same as in a full hard polytomous bush. (Recall that these statements concern one character; this procedure does not necessarily yield the dichotomous tree that has the greatest treelength over all characters.) For unordered characters, we have no proof, but with many examples we have been able to find a dichotomous tree requiring as many steps as the hard bush, and we suspect that there will always exist one.

MacClade does not have algorithms for the minimum conceivable numbers of steps for stratigraphic, Dollo, and user-defined characters, nor for the maximum for irreversible, stratigraphic, Dollo, and user-defined characters. For characters of user-defined types, the minimum and maximum conceivable number of steps is in general very difficult to calculate. For this reason the consistency index is given only for unordered, ordered, irreversible, and stratigraphic characters, whereas the retention index and rescaled consistency index are given only for unordered and ordered characters.

Early versions of PAUP 3.0 calculated the minimum conceivable number of steps differently from MacClade. See [Chapter 26](#) for details.

Consistency and retention indices

MacClade will calculate the consistency index (Kluge and Farris, 1969; Farris, 1989), retention index (Archie, 1989a,b [as HERM]; Farris, 1989), and the rescaled consistency index (Farris, 1989) for the tree as a whole and for the characters individually. Calculations for these indices are requested in the Σ menu. Indices for the tree are displayed in the tree legend; those for individual characters are displayed in the character list window.

Character consistency and retention indices

The consistency index (CI) for each character i is calculated as:

$$\text{Character CI} = \frac{m_i}{s_i}$$

The retention index (RI) for each character i is calculated as:

$$\text{Character RI} = \frac{M_i - s_i}{M_i - m_i}$$

The rescaled consistency index (RC) for each character is calculated as:

$$\text{Character RC} = (\text{character CI}) \times (\text{character RI})$$

where m_i is the minimum conceivable number of steps for character i on any tree, M_i is the maximum conceivable number of steps, and s_i is the number of reconstructed steps for character i on the particular tree in question (for instance, that on the screen). Thus the CI would be 1 if the character had no homoplasy, 0.5 if there were twice as many steps as needed, and so on. By convention MacClade indicates an index of 0 if both the numerator and denominator are 0 (e.g., for invariant characters, the CI is 0/0).

Ensemble consistency and retention Indices

The CI, RI, or RC of a collection of characters is an ensemble index (Farris, 1989). MacClade calculates ensemble indices over all included characters, and presents these in the tree legend as the CI, RI, or RC of the tree. MacClade also calculates ensemble indices in the chart window, whenever the index for a collection of two or more characters is requested.

The consistency index CI for all characters on a tree is the minimum possible treelength divided by the observed treelength. To be exact, it is the weighted sum of the minimum conceivable number of evolutionary steps for each character of the n characters divided by the weighted sum of the observed number of steps for each character:

$$\text{Tree CI} = \frac{\sum_{i=1}^n w_i m_i}{\sum_{i=1}^n w_i s_i}$$

where w_i is the weight applied to character i. Characters of stratigraphic, Dollo, or user-defined type and invariant characters are not included in the CI. If the characters in the data matrix are perfectly congruent with each other and the tree, and thus there is no homoplasy, then the observed number of steps will equal the minimum, and the CI will be 1.00. The messier the data on the tree, and the greater the amount of homoplasy, the greater is the observed number of steps, and the more CI shrinks. CI lies between 0 and 1. MacClade does not remove the contribution of autapomorphies of terminal taxa from the overall CI. It is the responsibility of the user to avoid inflating the CI with autapomorphies. If you want to remove uninformative characters from the consistency index, use **Exclude Uninformative** ([page 279](#)).

The retention index RI for all characters on a tree is calculated as the (maximum possible treelength –

actual treelength)/(maximum possible treelength – minimum possible treelength). To be specific:

$$\text{Tree RI} = \frac{\sum_{i=1}^n w_i M_i - \sum_{i=1}^n w_i s_i}{\sum_{i=1}^n w_i M_i - \sum_{i=1}^n w_i m_i}$$

Invariant characters, as well as characters of irreversible, stratigraphic, Dollo, or user-defined type, are not included in the RI. If the characters in the data matrix are parsimony-informative and perfectly congruent with each other and the tree, RI will have a value of 1. If the data are maximally homoplastic on the tree, RI will have a value of 0.

The rescaled consistency index RC for all characters on a tree is the CI multiplied by the RI. Characters of irreversible, stratigraphic, Dollo or user-defined type and invariant characters are not included in the RC. As with RI, RC ranges from 0 to 1, with higher RC values indicating that characters in the data set are more congruent with each other and the tree.

Notation used for indices

The **Index Notation** dialog box in the Σ menu allows you to alter how the consistency index, retention index, and rescaled consistency index are displayed. You can vary the number of significant digits displayed between one and four. You can also choose to have the indices displayed using exponential notation in the character list window.

Calculating the percentage of missing data and gaps

MacClade automatically calculates the percentage of missing data and gaps in each taxon, and displays this in the taxon list window.

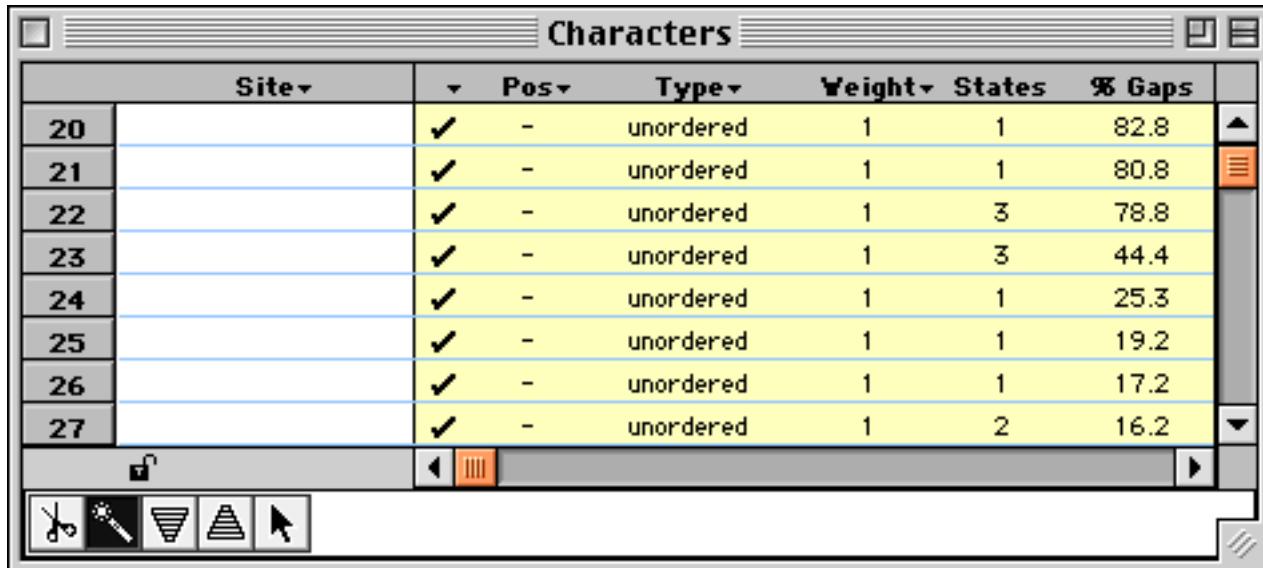
The screenshot shows the MacClade Taxa window. The window title is "Taxa". It contains a table with the following data:

| | Taxon | Seq Length | % Missing | % Gaps |
|---|-------------|------------|-----------|--------|
| 1 | Sialis | 1934 | 0.0 | 30.7 |
| 2 | Phaeostigma | 2140 | 0.0 | 23.4 |
| 3 | Oliarces | 2260 | 0.0 | 19.1 |
| 4 | Chrysoperla | 1185 | 0.0 | 57.6 |
| 5 | Anisochrysa | 2157 | 0.0 | 22.7 |

At the bottom of the window, there is a toolbar with several icons: a pair of scissors, a magnifying glass, a triangle, a square, and a cursor icon.

MacClade will also calculate the percentage of missing data or gaps in each character if you open the character list window (by choosing **Character List** in the **Characters** menu), then, in the **Char List Options** sub-

menu in the **Characters** menu, choose either **Show % Missing** or **Show % Gaps** or both. This will add one or two columns to the character list window, as shown below.

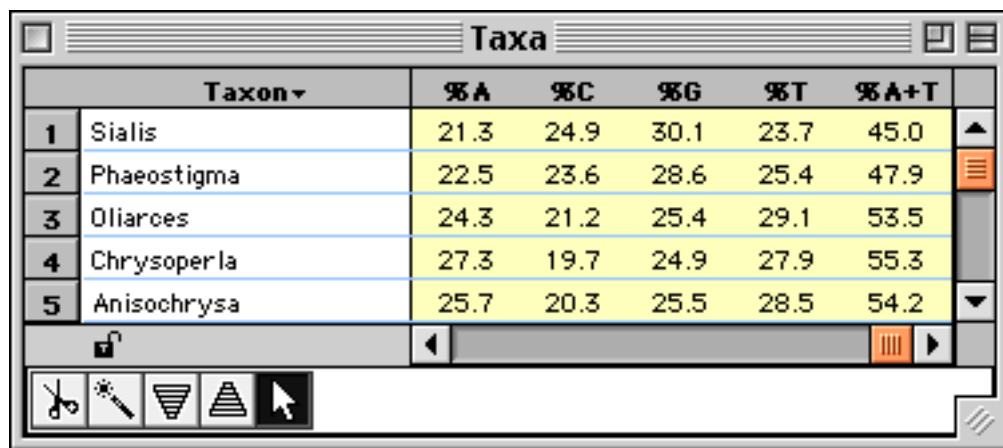


The screenshot shows the 'Characters' window in MacClade. The table has the following data:

| Site | Pos | Type | Weight | States | % Gaps |
|------|-----|------|-----------|--------|--------|
| 20 | ✓ | - | unordered | 1 | 1 |
| 21 | ✓ | - | unordered | 1 | 1 |
| 22 | ✓ | - | unordered | 1 | 3 |
| 23 | ✓ | - | unordered | 1 | 3 |
| 24 | ✓ | - | unordered | 1 | 1 |
| 25 | ✓ | - | unordered | 1 | 1 |
| 26 | ✓ | - | unordered | 1 | 1 |
| 27 | ✓ | - | unordered | 1 | 2 |

Calculating nucleotide percentages for each sequence

The percentage of As, Cs, Gs, and Ts in each sequence will be displayed in the taxon list window if you choose **Show Nucleotide Frequencies** from the **Taxon List Options** submenu of the **Taxa** menu:



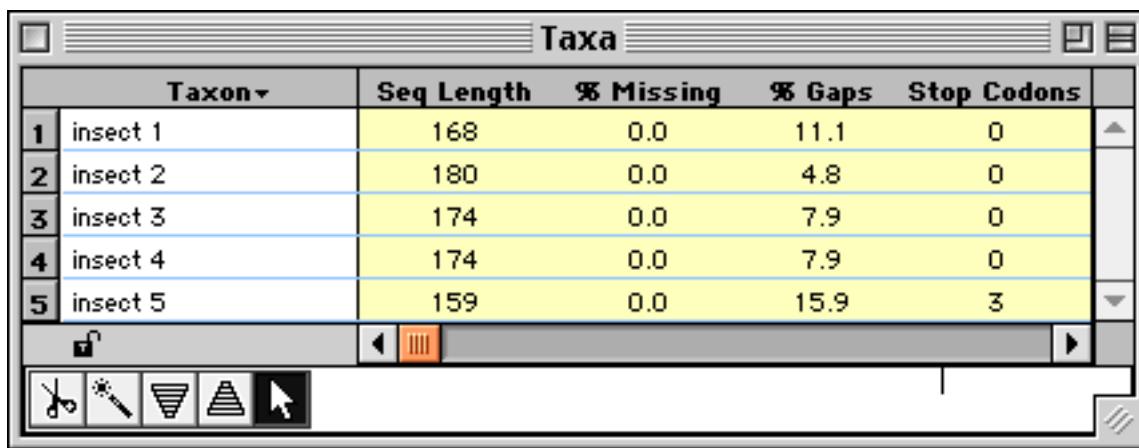
The screenshot shows the 'Taxa' window in MacClade. The table has the following data:

| Taxon | %A | %C | %G | %T | %A+T |
|---------------|------|------|------|------|------|
| 1 Sialis | 21.3 | 24.9 | 30.1 | 23.7 | 45.0 |
| 2 Phaeostigma | 22.5 | 23.6 | 28.6 | 25.4 | 47.9 |
| 3 Oliarces | 24.3 | 21.2 | 25.4 | 29.1 | 53.5 |
| 4 Chrysoperla | 27.3 | 19.7 | 24.9 | 27.9 | 55.3 |
| 5 Anisochrysa | 25.7 | 20.3 | 25.5 | 28.5 | 54.2 |

To see nucleotide percentages over multiple sequences at once, use the States chart ([page 404](#)) or Trace All States ([page 371](#)).

Calculating the number of stop codons in protein-coding nucleotide sequences

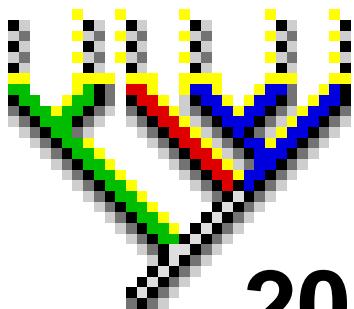
In protein-coding nucleotide sequences, the number of stop codons present in each sequence, as defined by the current genetic code and the current reading frame, will be displayed if you choose **Show # Stop Codons** in the **Taxon List Options** submenu of the **Taxa** menu:



The screenshot shows the 'Taxa' window in MacClade 4. The window title is 'Taxa'. The table has columns: Taxon#, Taxon name, Seq Length, % Missing, % Gaps, and Stop Codons. The data is as follows:

| Taxon# | Taxon | Seq Length | % Missing | % Gaps | Stop Codons |
|--------|----------|------------|-----------|--------|-------------|
| 1 | insect 1 | 168 | 0.0 | 11.1 | 0 |
| 2 | insect 2 | 180 | 0.0 | 4.8 | 0 |
| 3 | insect 3 | 174 | 0.0 | 7.9 | 0 |
| 4 | insect 4 | 174 | 0.0 | 7.9 | 0 |
| 5 | insect 5 | 159 | 0.0 | 15.9 | 3 |

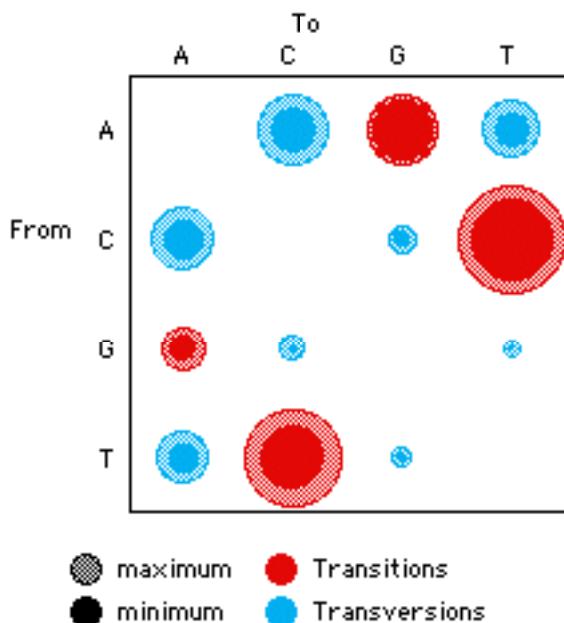
This option can be useful for manual sequence alignment.



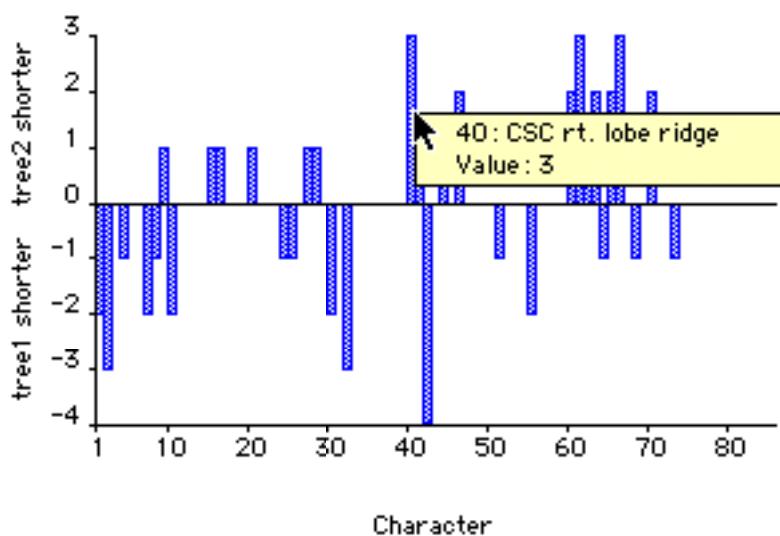
CHARTING TREE AND CHARACTER STATISTICS

20

Various statistics concerning trees and characters can be plotted as bar charts, bubble charts, or tables. The charts can summarize, among other statistics, the numbers of steps, or the types of changes reconstructed for one or more characters. The charts can be based on just the tree shown on the screen, or on multiple trees stored in tree files, or on randomly generated trees. Two example charts are shown below.



A State Changes &
Stasis chart showing
the relative frequen-
cies of reconstructed
changes



A Compare 2 Trees
chart showing the
minimum difference in
numbers of steps
between two trees

The charting features are available in the **Chart** menu. Examples of the use of various sorts of charts are found both in this chapter, and in [Chapter 16](#).

General issues concerning chart calculations

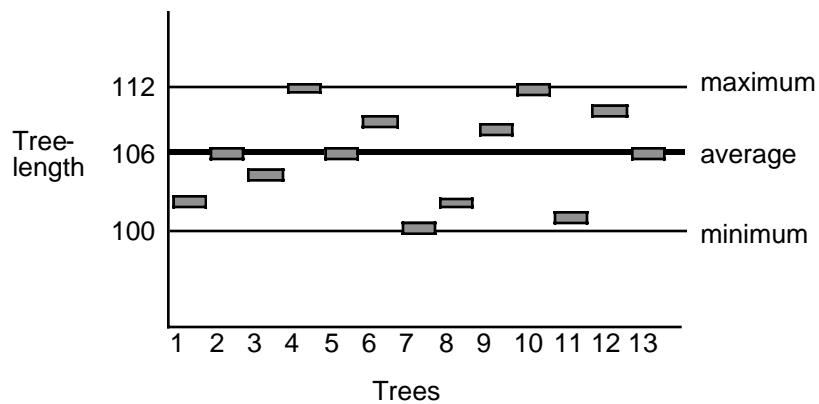
This section discusses several issues that are relevant to the calculations and interpretation of most charts. For commands that allow you to alter the display of a chart, see the section "[Chart display](#)" toward the end of this chapter.

Multiple trees

In any inference of phylogeny, rarely is there only one acceptable tree. More often, there are multiple trees that are well supported by the data. One of the main aims of the charting facilities in MacClade is to make easier the examination of multiple trees, so that you can see how the trees vary in their implications about character evolution.

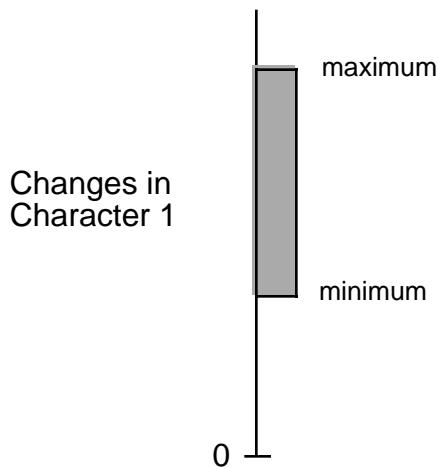
In general, for those charts that summarize features of multiple trees, trees can come either from stored trees (in the data file or external tree file if one is active), from a different external tree file, or from MacClade's random tree generator.

Different trees will often differ in the values of the statistics calculated. For example, over 13 trees, there might be variation in treelength between 100 and 112:

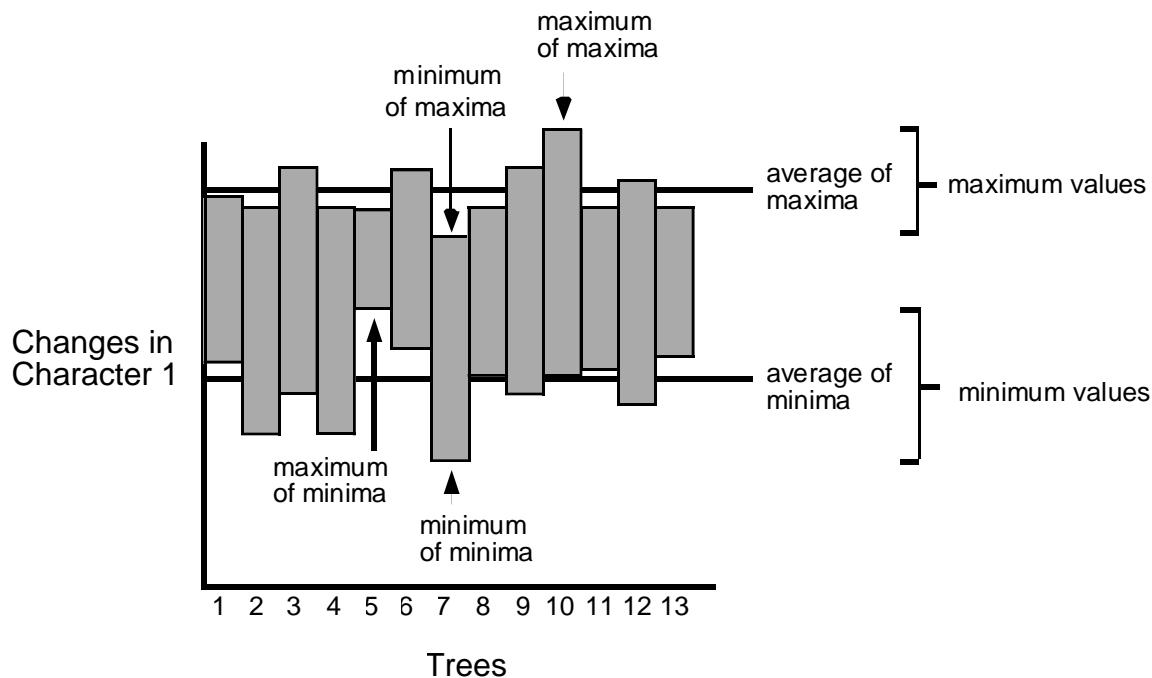


For treelength, each dichotomous tree will have a single value. Over all trees, there will be a minimum value (in this example, 100), a maximum value (112), and an average value (106).

For other statistics, each dichotomous tree may have more than a single value. For example, some of many equally parsimonious reconstructions of character evolution may involve several changes for character 1, whereas other reconstructions will involve few changes. If this is true, then MacClade will store the minimum and maximum values for that tree.



Thus, over multiple trees, there will be a minimum value of the minimum values, and average of minima, and a maximum value of maxima; there will also be minimum, average, and maximum values of the maxima:



In this circumstance, there are six possible values MacClade could display to you: the minimum, average, or maximum of the minima or maxima. With this diversity of inferred values, MacClade can't display them all at once, but MacClade allows you to choose various subsets of these six values to be displayed (using the **Values to Show** dialog box in the **Display** menu; see [page 406](#)).

WARNING: *The interpretation of average values is problematical, as these are averages of a series of alternative hypotheses (e.g., different trees), not averages of a set of data values. Unless one is willing to assign equal probabilities to the alternative hypotheses, these averages are thus best thought of as descriptors of acceptable hypotheses, rather than estimates.*

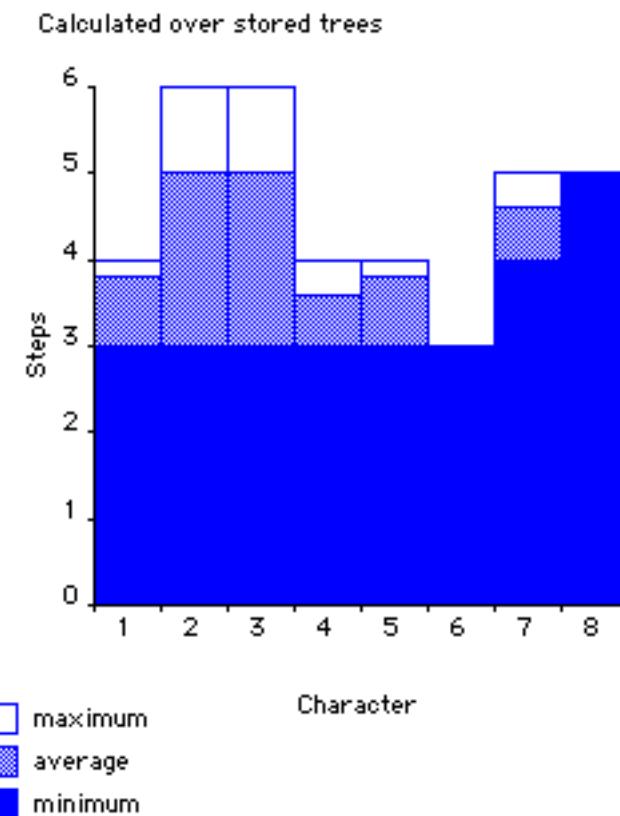
NOTE: If you wish to make a chart from trees in a tree file, but do not need to open the tree file for any other purpose, then you will save memory by requesting the tree file from within the charting facility, rather than by first choosing **Open Tree File** from the **Trees** menu.

NOTE: The **Compare 2 Tree Files** chart reads the tree files off of the disk, even if one of the tree files is the tree storage currently in use by MacClade's tree window (whether an external tree file or the data file itself). If you request that the chart use the current tree storage file, the copy of the file on disk that is used will not contain any trees that you have recently asked MacClade to store, unless you have subsequently saved the file. These stored but unsaved trees will not be included in chart calculations.

Text files with values for each tree

When MacClade calculates statistics over multiple trees, in many charts only the minimum, maximum, and average values over those trees are presented graphically. If you wish to know *all* of the values for each tree, not just the ranges, select the "log file" check box available in the dialog box that comes up when you choose the sort of chart to plot. MacClade will generate a text file with all of these values.

For example, imagine that you have a small data file with 8 characters, and you wish to examine how the number of steps for each of these characters varies over a collection of 5 stored trees. The chart presented by MacClade might look like the example chart below.



Here we see that there is some variation between the trees. For example, the first character varies between 3 and 4 steps across the five trees. If we asked MacClade to write a text file detailing the values for each tree, by checking "log file" in the **Chart Options** dialog box, the text file would look something like that shown below.

```

MacClade version 4.0
Sunday, 9 July 2000: 1:54 PM

Data file: Example File

Steps in characters in stored trees

Tree source: trees in data file examined.

One column per character; one row per tree

tree #    name      1   2   3   4   5   6   7   8
1        PAUP.1    4   5   5   3   4   3   4   5
2        PAUP.2    4   3   5   4   3   3   5   5
3        PAUP.3    4   5   3   4   4   3   4   5
4        PAUP.4    3   6   6   3   4   3   5   5
5        PAUP.5    4   6   6   4   4   3   5   5

```

The first column in the table is the tree number, the second is the tree name, and the other eight columns contain the number of steps in each of the eight characters.

Some of the values written to these text files may be very large or very small, and MacClade may write these in exponential notation (e.g., 1,200,000 would be written 1.2 E 6). If you wish to edit these text files, and then read them in to some other program (perhaps a statistics application), you should make sure that your other program can read exponential notation. If not, you may wish to force MacClade to avoid exponential notation. This can be done by making sure "use exponential notation in output files" is *not* selected in the **Other Options** dialog box in the **Options for Saving** submenu of the **File** menu.

MacClade can also write a text file of chart calculations for **State Changes & Stasis** charts calculated over a single tree if minimum-maximum changes are requested.

Forbidden trees in chart window

When you request a chart summarizing features of multiple trees stored in the data file or a tree file, MacClade may find that some of the trees have features forbidden in the current situation.

Trees with polytomies or observed taxa in ancestral position will **not** be allowed if:

1. Explicit or implicit evaluation of all MPRs is needed (for Character Steps/etc. chart with changes option specified; for State Changes & Stasis chart with average or minimum/maximum option),
2. Any characters being analyzed are of Dollo or user-defined types.

Additionally, trees with polytomies are not allowed if any characters being analyzed are of irreversible or stratigraphic types.

Should MacClade find any trees with forbidden features, it will warn you. If the forbidden features are polytomies, MacClade will give you the option to (1) ignore all trees with polytomies or (2) choose a random dichotomous resolution for any polytomies encountered (see [Chapter 23](#)). If the forbidden feature is observed taxa fixed in ancestral position, MacClade gives you no choice; it automatically ignores these trees.

Polytomous vs. dichotomous trees in chart window

Even if polytomous trees are not forbidden, there may be reasons to avoid them, as discussed in [Chapter 4](#). MacClade's chart window provides various ways to randomly sample from dichotomous resolutions of the tree, so as to avoid some of the problems of polytomies. This random sampling can be done either (1) in response to a request by MacClade after it has found forbidden polytomies, as discussed above, (2) by direct request of the user before any calculations start, using the Resolve Polytomies button in many of the chart dialog boxes, or (3) by asking for a special option of random trees that resolves the tree on screen randomly. When the Resolve Polytomies button is used and the choice is made to resolve polytomies, MacClade will resolve randomly any polytomies in processed trees.

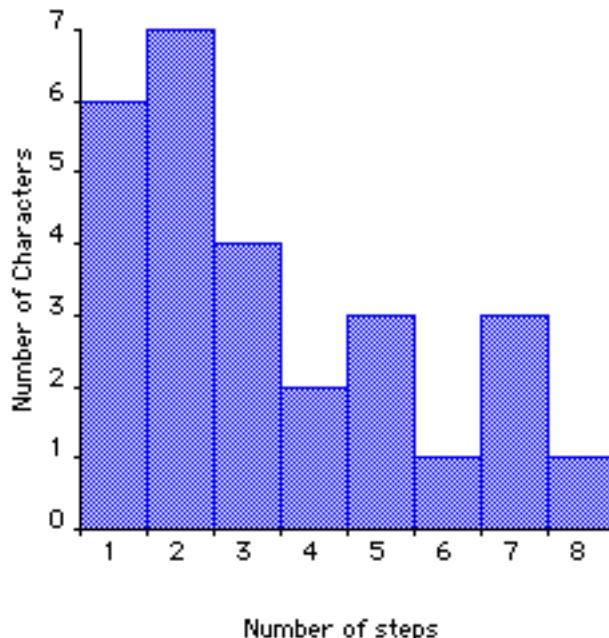
Interval widths in bar charts



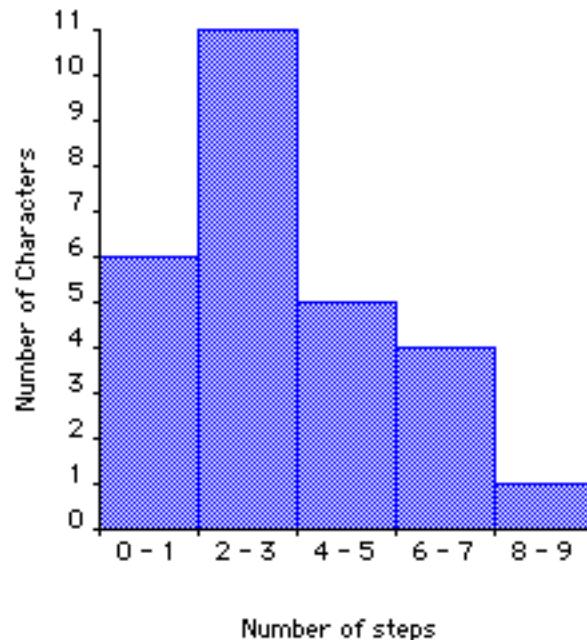
For charting functions that yield bar charts, the interval width button () appears in the **Chart Options** dialog box. If you click on this button, a dialog box will appear in which you can choose how values are grouped on the horizontal axis of the bar chart, and the maximum number of horizontal intervals.

For instance, if the chart window is to display a graph showing the number of steps in each character arranged in sequence, the default is to have one vertical bar for each character. However, by setting the bar chart interval width to 3 characters, the chart will sum every three characters, so that the first vertical bar is for characters 1–3, the second 4–6, and so on. Changing the value of the interval width will cause the entire chart to be recalculated.

As another example, consider a chart in which the number of steps per character is on the horizontal axis. By default, each bar will have a width of 1 step:



If you reset the interval for 2 steps, then the first interval will be for characters with 0-1 steps, the second interval 2-3 steps, and so on:



If the horizontal axis lists character numbers, and the statistic on the vertical axis is CI, RI, or RC, and two or more characters are grouped together for each bar of the chart, then the appropriate ensemble index ([Chapter 19](#); Farris, 1989) will be calculated for the characters represented by each bar.

When MacClade recalculates charts

If you have a chart window on the screen, and you change a background assumption of the chart, MacClade will recalculate the chart. For example, a chart will be recalculated if the chart is for the current tree only, and you have changed the current tree, or if the chart is for the traced character only, and you have changed the type of the character, or changed the fixing of ancestral states. There is at least one exception; namely, if a chart was based on a tree file and that tree file was modified by deleting or storing trees, the chart is not recalculated. There may be other apparent exceptions, but they are not really exceptions. For example, changing the current tree does not affect a multiple trees chart even if the current tree had been read from a tree file, because you are not actually changing the tree file; or, fixing the state of a traced character affects State Changes & Stasis chart if it is for the current tree, but does not affect the chart if it is for multiple trees, because this chart is not based on the current tree or its fixing.

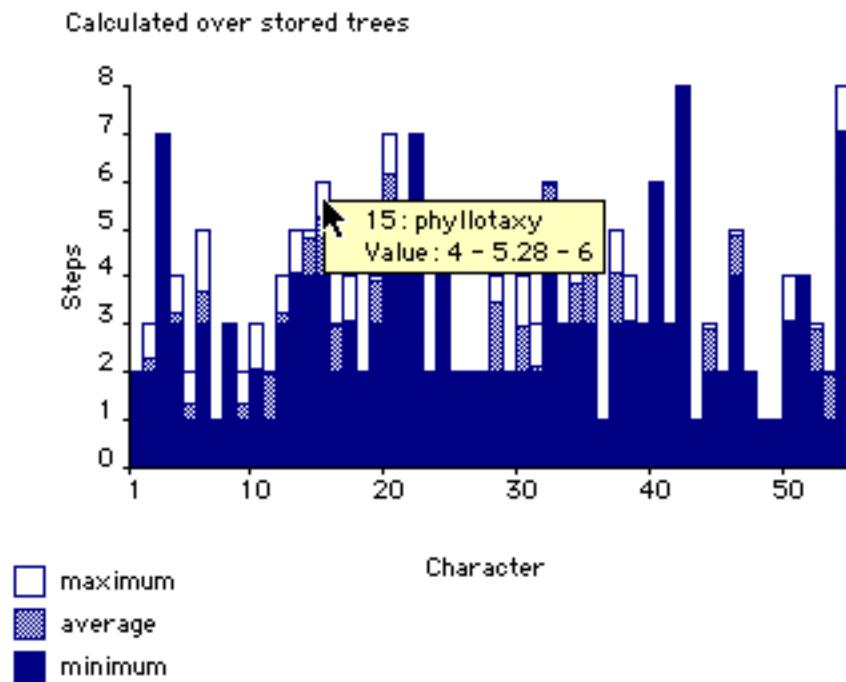
If the chart was based on multiple trees and you change a background assumption, MacClade will give you the choice of updating the chart or closing it. This allows you to avoid delays if you no longer want to keep the chart up-to-date. If **Quietly Recalculate** in the **Chart** menu is checked, then MacClade will recalculate charts without asking.

Changing the display of the chart by using the items in the **Display** menu will not cause the chart to be recalculated, but calling forth the **Chart Options** dialog box and pressing the **Chart** button will.

Determining the categories and values on a chart

For some bar charts and bubble charts, it can be difficult to determine what category a particular bar or bubble is referring to, or what the value is for a particular category. Clicking on the part of the bar or bub-

ble chart in question will present a brief description of the category and associated value, as shown below.



Grayed horizontal axes

If a portion of the horizontal axes is dimmed or grayed, with no bars shown for these categories, then that means that those categories are not applicable for the current chart (e.g., if characters are plotted along the horizontal axis and some characters are excluded).

Making the chart window frontmost

If you have asked MacClade to calculate and display a chart, but the chart window has subsequently become hidden by another of MacClade's windows, you can bring the chart window to the front again by choosing **Chart** from the **Windows** menu.

Printing and saving charts

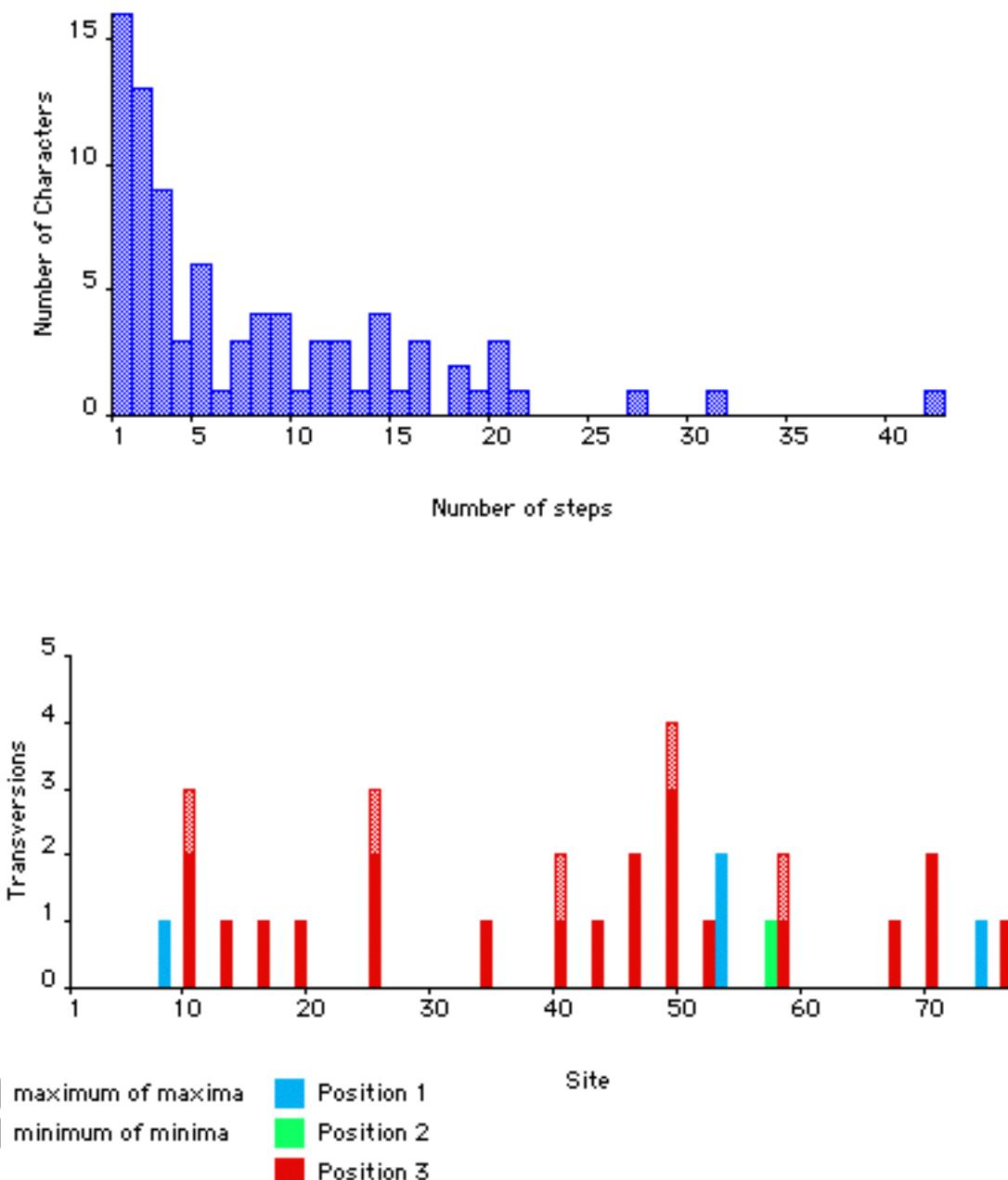
Printing charts and saving charts as graphics and text files are described in ["Printing charts" on page 464](#).

Character Steps/etc.

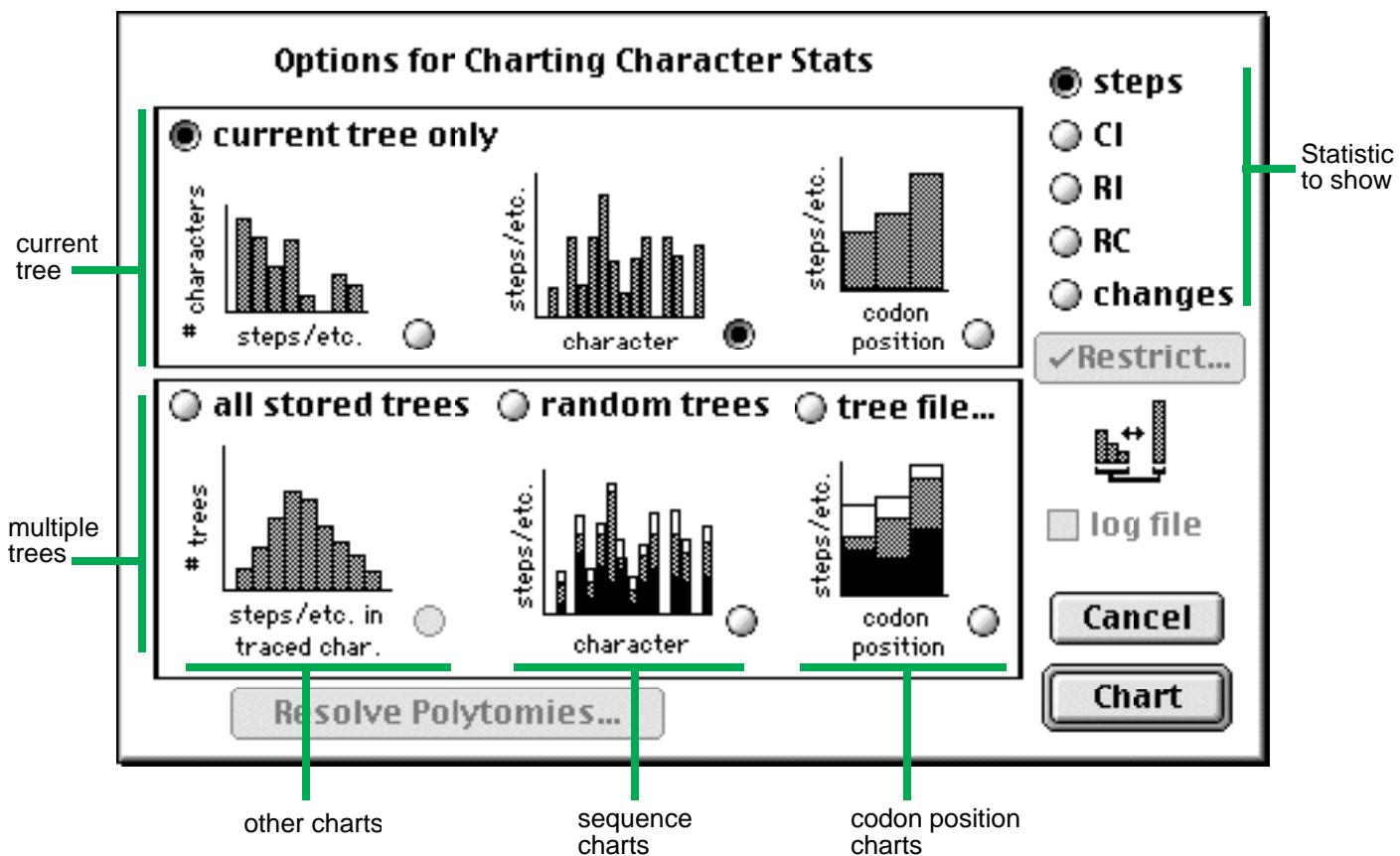
Charts concerning the number of steps in characters and related statistics about characters are available under the **Character Steps/etc.** item in the **Chart** menu. The related statistics include consistency index, retention index, rescaled consistency index, and the number of changes. For each of these statistics, you can make various charts concerning the current tree or multiple trees that are stored, read from a file, or randomly constructed, and summarizing all included characters or only the currently traced one. The current data and assumptions are used to calculate these statistics.

For example, you could be interested in seeing the distribution of the steps in a character over stored trees, or the distribution along a DNA sequence of the number of transversions on the current tree. These are

shown, respectively, in the following two charts.



Options for charting statistics about characters are chosen in the following dialog box:



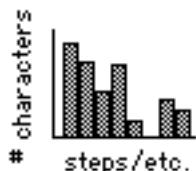
Statistic displayed: Steps, index, or changes

On the right of the above dialog box are listed the choices for the statistic displayed: the number of steps required by the tree for a character, the consistency index, retention index, rescaled consistency index, or the number of changes required in parsimonious reconstructions of character evolution. The meaning of each of these is described in [Chapter 19](#).

If the statistic displayed is the number of steps, then it will be weighted by character weights, except if the chart displays the number of steps in the traced character over multiple trees, in which case the steps are unweighted.

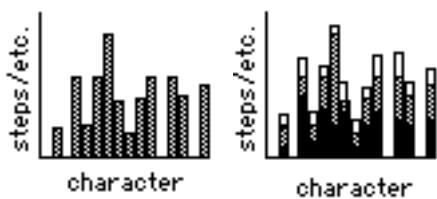
If the statistic displayed is the number of changes, MacClade may indicate a range of numbers of changes for a single character on a single tree. The range indicates the minimum and maximum number of changes allowed in the various parsimonious reconstructions of character evolution (see "[Ancestral states and reconstructions](#)" on page 67). When changes are shown, you can ask that only certain changes be shown. For instance, you might ask to show only transitions or only transversions for DNA data. The method to do this is described below under "[Restricting changes to those of a given class](#)" on page 393.

Number of characters with given number of steps/etc.



This option displays a bar chart showing the number of characters with given numbers of steps, consistency indices, and so on, on the current tree. Thus, for instance, you might ask for the number of characters with various values of the consistency index. If characters tend to be bimodal, either agreeing strongly with the tree or strongly against, this might lead to a different interpretation about the evidence for the tree than if the characters had had a more mixed set of consistency indices. If the statistic displayed is the number of changes, then there may be two superimposed bar charts, one for the minimum number of changes, the other for the maximum number.

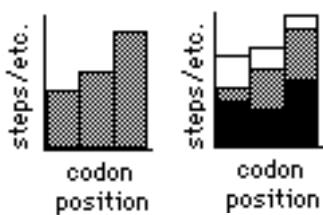
Number of steps/etc. on each character in sequence



This option displays the number of steps, consistency indices, and so on, in each of the characters in sequence. The vertical axis is the number of steps or other statistic; the horizontal axis is the character number. This option is especially useful to see non-random distributions of change in molecular sequence data (see [Chapter 16](#)).

You can ask MacClade to reweight the characters based upon the values in this chart (see "[Creation of weight sets based on calculated indices for each character](#)" on [page 276](#)).

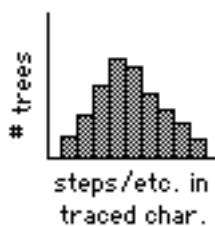
Number of steps/etc. by codon position



For DNA or RNA sequence data, you can chart by codon position. If this is chosen, then MacClade will sum all steps or changes in every first, second, and third codon position. If CI, RI, or RC are requested, then MacClade calculates the ensemble values ([Chapter 19](#); Farris, 1989) for these indices over all first positions, second positions, and third positions. You might expect to see, for instance, that the number of steps in third positions is larger than in first and second positions in protein-coding regions. See "[Specifying coding regions and codon positions](#)" on [page 292](#) for a description of how codon position is defined and the options for altering the definition.

You can ask MacClade to reweight the characters based upon the values in this chart (see "[Creation of weight sets based on calculated indices for each character](#)" on [page 276](#)).

Number of trees with given number of steps/etc.



This option displays the number of trees, either stored in the data file or an external tree file or created randomly, that have various values for the statistics as they apply to the currently traced character. Thus, for instance, you might see how the traced character differs in its number of steps across the stored trees.

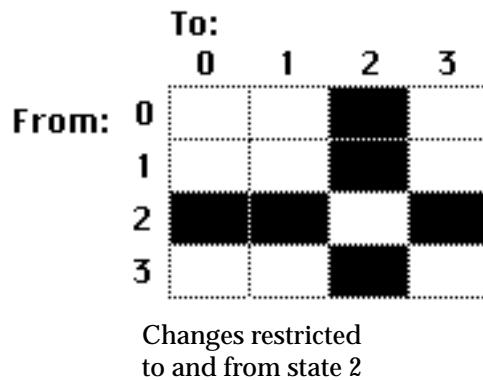
NOTE: If you hold down the Option key as you press the Chart button in the **Character Steps/etc.** dialog box, and you request the number of steps in a traced character over multiple trees, and the character is of type unordered, ordered, irreversible, or stratigraphic, and text file is requested, then MacClade will also will write into the text file a list of the states assigned to the root branch for each tree examined.

Restricting changes to those of a given class

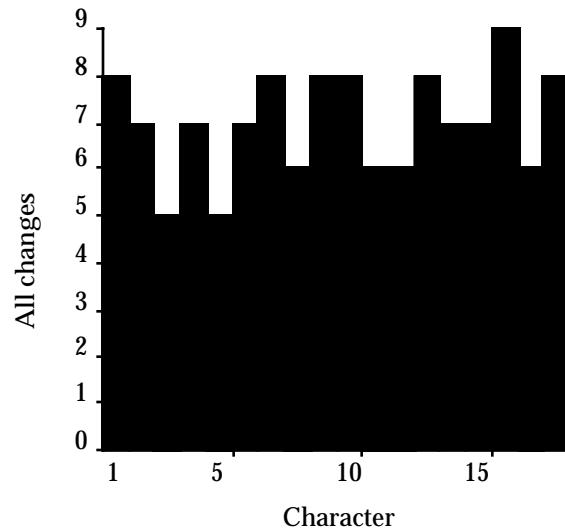
If the statistic you are asking MacClade to graph is changes, by default MacClade will count or show all changes that are reconstructed. If you are not interested in all changes, but only changes of a particular sort, you can ask MacClade to include only changes of a particular class.

To do this, you first select changes as the index to be charted in the **Character Steps/etc.** dialog box, and then press the Restrict button. You will be presented with a dialog box, in which you select the sorts of changes you wish to include; you can also give a name to the class of changes you select.

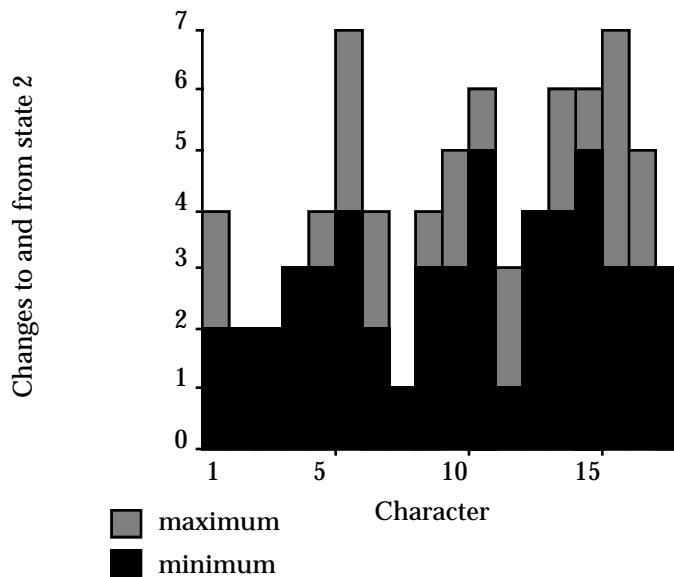
For example, you could ask MacClade to chart only those changes to or from state 2, as shown below:



In the "Restrict Changes Example" file included with MacClade, if you ask to show a Character Steps/etc. chart, plotting changes over the current tree for each character, you will get a graph like the following:



This graph shows all changes over each of the 18 characters. If you restrict it to just changes to or from state 2, you can see which of the changes in the above chart were changes to or from state 2:

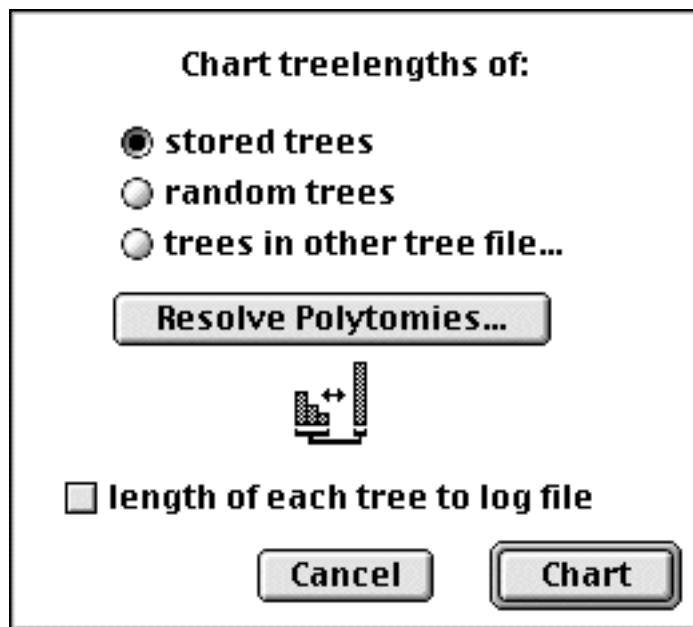


Because different equally parsimonious reconstructions of the evolution of single characters vary in the number of changes to and from state 2, there is a minimum and maximum value for many of the bars.

If your data are DNA or RNA sequences, then you can easily restrict changes to transversions or transitions, so that you can examine, say, the distribution of transversions along a DNA sequence.

Treelengths of many trees

The **Treelengths** chart makes a bar chart or table of the numbers of trees with various numbers of treelengths. The dialog box in which you can choose options is shown below.



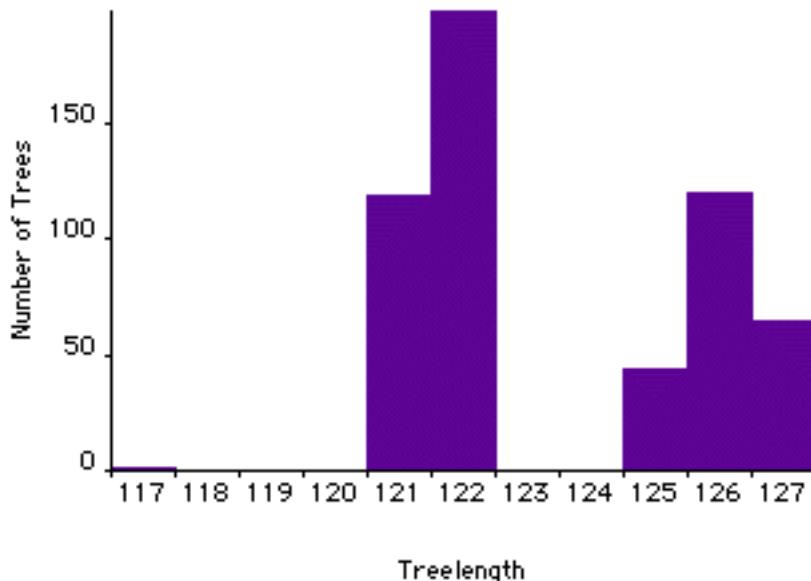
If the treelengths are charted over the stored trees in the current tree repository, the tree currently shown on screen is not included unless it has been explicitly stored. The current data and assumptions are used to

calculate the treelengths.

Treelengths can also be charted over a set of randomly generated trees (including random resolutions of a displayed polytomous tree). If you choose this option, MacClade will present you with the dialog box to generate random trees (see [Chapter 23](#) for a description of the **Random Trees** dialog box).



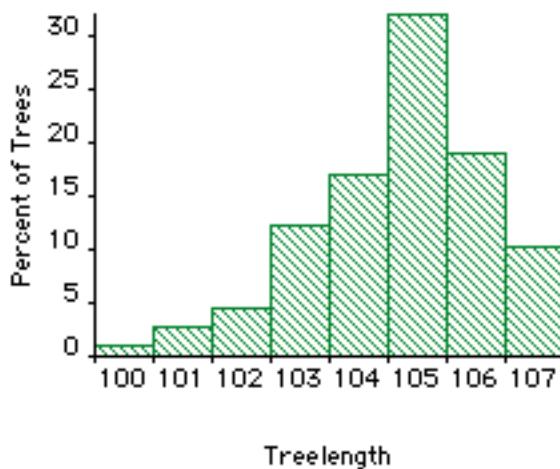
EXAMPLE: The "Bembidion" example data file in your Examples folder contains a matrix of structural and cytogenetic characters for a group of ground beetles (D. Maddison, 1993). There exist 545 equally parsimonious trees for this data matrix, if one considers all adult and larval characters included in the matrix. The trees are stored in the file "Bembidion.trees". You can examine whether there is variation in how well the larval characters alone fit onto the 545 trees with MacClade's chart of treelengths. Open the "Bembidion" data file, go to the tree window, and choose the "larval only" inclusion set using the **Inclusion Sets List** window from the **Inclusion Sets** submenu in the **Characters** menu. Then ask MacClade to chart treelengths. In the dialog box that is presented, choose "trees in other tree file...", and select the tree file "Bembidion.trees". Then click on the **Polytomies** button; in the dialog box that is presented, ask MacClade to resolve all polytomous trees; click **OK** on this dialog box. Then press **Chart** in the **Treelengths** dialog box. You should be presented with the following chart:



Note the one tree at length 117. Clearly, there is one tree among the 545 that is much more favored by the larval characters than the other 544 trees. This of course implies that there is more homoplasy on this one tree in adult characters than there is in the other 544 trees.



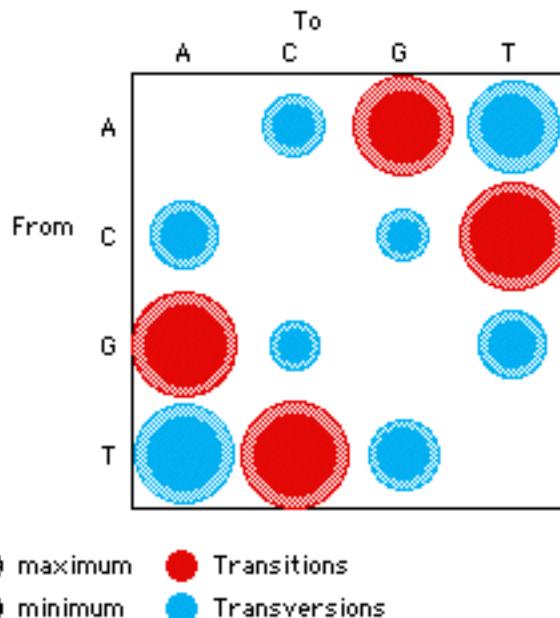
EXAMPLE: Open the file "Polytomy Example". Go to the tree window. Note that the length of the tree is given as 91+ if the polytomy is presumed to represent uncertain resolution, and 109 if it is presumed to represent multiple speciation. (You can switch between these two modes using the **Polytomy Options** item in the **Trees** menu.) Now ask MacClade to chart the treelengths of random trees. In the dialog box that is presented, in which you are to specify the type of random tree, choose "randomly resolve current tree", and ask for a large number of trees. Choosing 100,000 random resolutions of the polytomous tree will yield a chart like the following:



This is a more accurate picture of the treelength of a polytomous tree than the treelength of 91+ or 109. (As there are only 105 resolutions of this polytomous tree, clearly some of the resolutions are being sampled more than once, but with a large enough sample they should be sampled more or less equally.)

State Changes & Stasis

The State Changes & Stasis chart shows the number of reconstructed state *i* to state *j* changes in a table or "bubble" chart to give an indication of the relative frequency of various changes over all included characters or just the traced character, and over the current tree or multiple trees. An example chart is shown below. A brief introduction to MacClade's bubble charts is given in ["Parsimony methods for inferring character change models" on page 60](#). The discussion of counting changes in ["Changes" on page 102](#) should also be consulted.

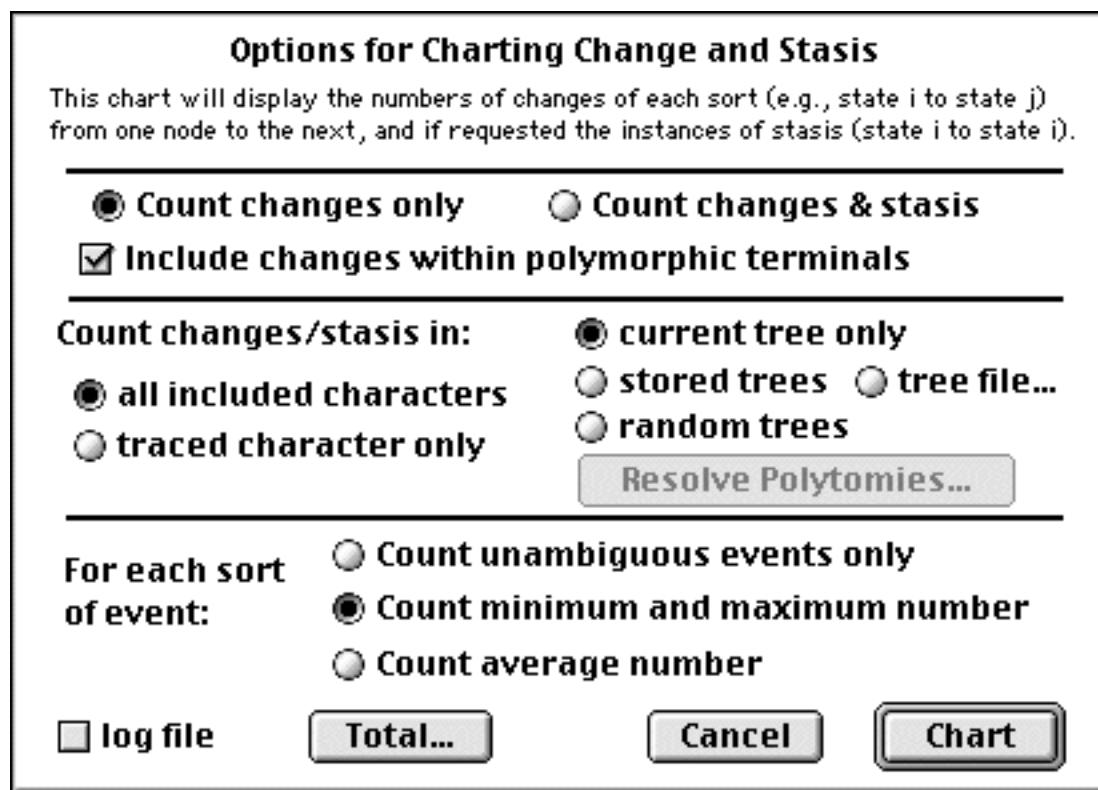


Ancestral states are reconstructed for all included characters. Each branch in the tree is then visited and

checked to see whether or not an *i* to *j* change is reconstructed as having occurred, either in one character or all characters depending on the option chosen. After MacClade visits all branches, it presents a matrix or bubble chart showing the total number of *i* to *j* changes in the tree for various *i* and *j*. (Actually, MacClade looks at all characters for one branch before moving to the next branch under the "unambiguous only" option, but it looks at all branches for one character before moving on to the next character under the "minimum and maximum" option.)

You can ask MacClade to create step matrices based upon the values in the Changes & Stasis chart; see ["Creation of a transformation type based on reconstructed frequency of changes" on page 288](#) for details.

When **State Changes & Stasis** is selected, a dialog box is presented to give various choices:



In the dialog box, you can choose which characters and trees to examine, and what sorts of events to show.

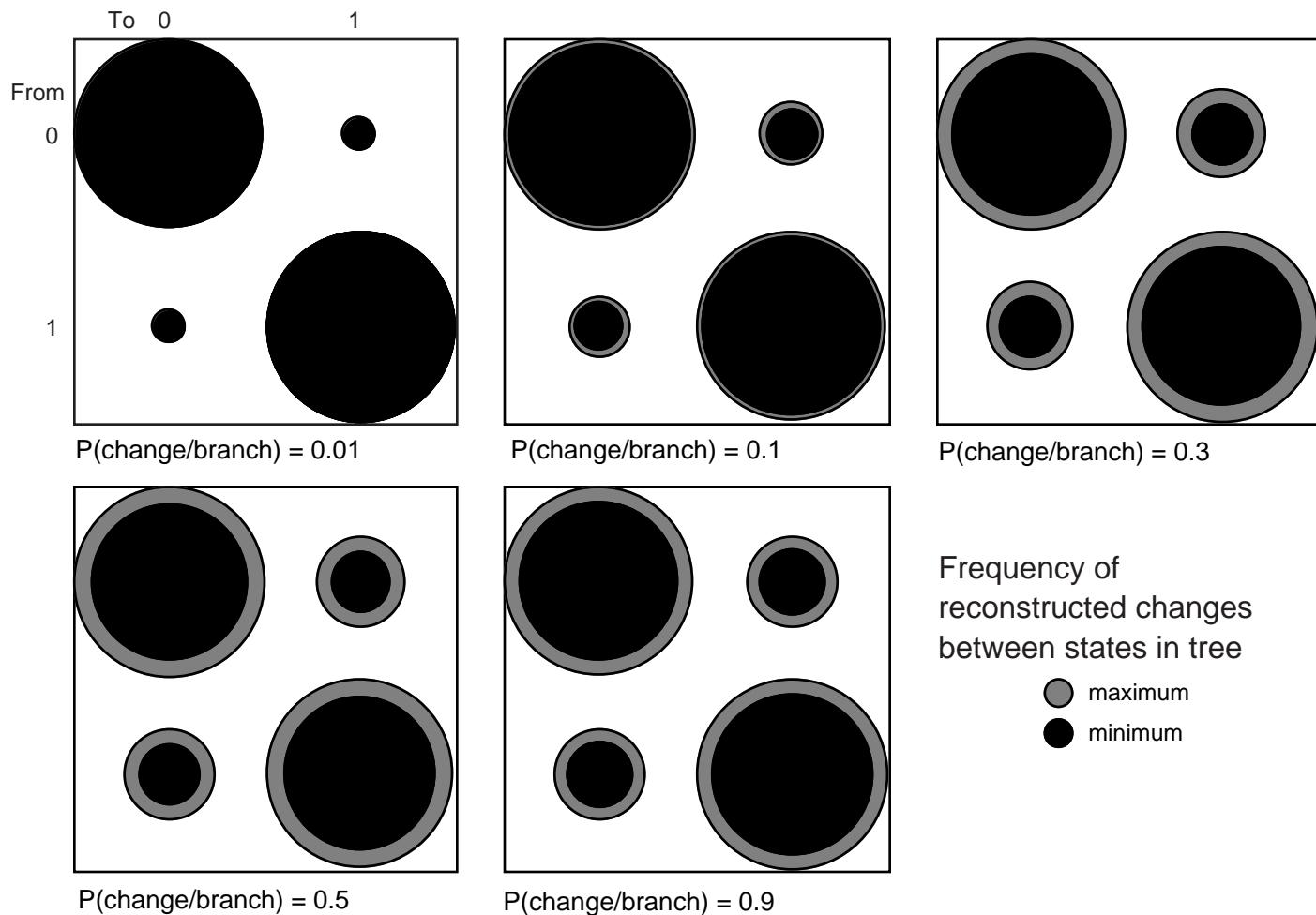
Reconstructions used in State Changes & Stasis charts

The State Changes & Stasis chart counts events of change and stasis based on a reconstruction of character evolution. The reconstruction used is the same sort of parsimony reconstruction of ancestral states shown in character tracings (see [Chapter 4](#) and [Chapter 18](#)). It therefore depends on the tree, the distribution of character states, and the transformations types assigned to the characters (see [Chapter 4](#)).

The reconstructions used are not the ACCTRAN/DELTRAN reconstructions, even when one of these is the chosen option in the **Resolving Options** dialog box, except when the chart is restricted to the current tree and traced character only. The reason for this is that ACCTRAN/DELTRAN is used only for the character tracing displayed on the tree. Likewise, if states are fixed at branches in the current tree, this affects the chart only if it is restricted to the current tree and traced character, because only the character tracing in the tree window is affected by fixing states.

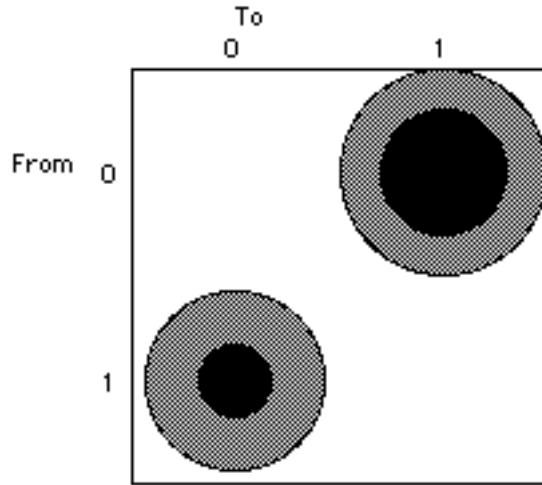
Users should be always aware that the reconstruction used as a basis for the State Changes & Stasis chart is only a reconstruction, and indeed is a parsimony reconstruction. Among other things, this means that the reconstruction will always yield a minimum estimate of the total amount of change.

EXAMPLE: The difficulty that parsimony has in reconstructing change when evolutionary rates are high can be illustrated using MacClade's facility to evolve characters stochastically up the current tree (described in [Chapter 23](#)) and the State Changes & Stasis chart. 1,000 binary characters were simulated on a tree of 20 taxa in each of five cases, differing in the probability of change on each branch: 0.01, 0.1, 0.3, 0.5, and 0.9. For each case, parsimony was used to reconstruct the changes, and the results summarized in a State Changes & Stasis chart, as shown below. Note that as the probability of change rises to 0.3 or more, parsimony cannot detect these higher probabilities: even when each branch has a 0.9 probability of a change, parsimony's reconstruction minimizing change shows changes on only about 3 out of 10 branches.

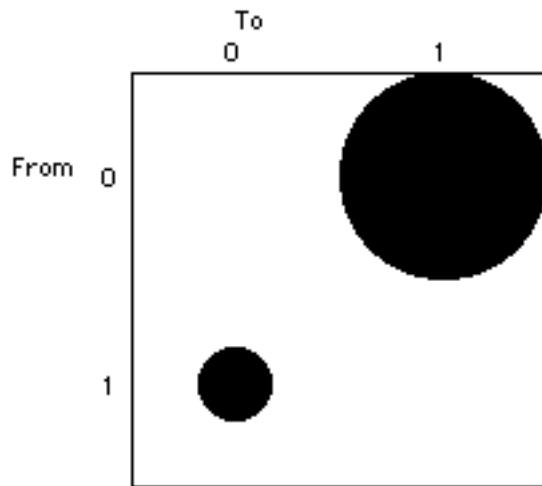


 **EXAMPLE:** The trees displayed and analyzed in MacClade are rooted trees. If you wish to examine an unrooted tree (that is, a tree in which the root is not specified), you need to follow several steps. An unrooted tree is a kind of consensus tree, consisting of all trees that represent possible rootings of the tree. For example, in the file "Unrooted Chart Example", the tree can be rooted along 17 different branches. If you wished to analyze an unrooted tree, say, for relative frequencies of 0 to 1 and 1 to 0 changes, you should examine all 17 rootings (D. Maddison, 1990). You can ask MacClade to do this by first choosing **New Tree File** in the **Trees** menu, then choose **All Rootings** from **Create Trees** submenu in the **Trees** menu. MacClade will store in the tree file 17 trees, representing each of the 17 possible

rootings. If you then ask for a State Changes & Stasis chart of all stored trees, counting minimum and maximum number of changes, you should get a bubble chart like this:



The size of the gray areas indicates that the rootings differ in their relative frequencies of 0 to 1 and 1 to 0 changes. This is in contrast to the chart you would see for just the current, rooted tree:



Change vs. stasis

If "Count changes and stasis" is requested, then instances in which a state does not change from one node to the next (i.e., the state at a node and its descendant node are the same) are included in the chart. Otherwise, only changes are included.

Changes in terminal polymorphisms

If "Include changes within polymorphic terminals" is selected, implicit changes within polymorphic terminal taxa will be included in the chart. See [Chapter 4](#) and [Chapter 18](#) for details on how MacClade does this.

Character traced vs. all characters

If "traced character only" in the **Chart Options** dialog box is selected, then only the traced character is

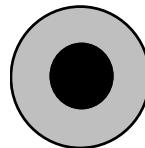
examined; if "all included characters" is selected, then all included characters are examined and the total number of i to j changes in all characters on the branches is counted. A discrete-valued character must be traced on the tree before "traced character only" can be selected.

Unambiguous vs. minimum/maximum vs. average

These choices are needed to deal with ambiguity in the reconstruction of character evolution. If several states can be placed at an ancestral node with equal parsimony, then exactly which changes occur may be ambiguous. For instance, there may be two equally parsimonious alternatives: two parallel 0 to 1 changes, or a 0 to 1 change followed by a 1 to 0 change.

When "unambiguous" is selected, an i to j change is counted as occurring on a branch only if parsimony requires i to be at the ancestral node below the branch, and j to be at the descendent node above the branch. Thus an i to j change is counted only when the ancestral and descendent nodes unequivocally have states i and j , respectively. In some situations there may definitely be an unambiguous change from 0 to 1, but where it occurred might be ambiguous. If the change could have occurred on a number of different branches, then it is not counted in the total number of unambiguous 0 to 1 changes. Thus, an i to j change is counted as unambiguous when it occurs on the same branch in all most-parsimonious reconstructions of character evolution.

When "minimum and maximum" is selected, all most-parsimonious reconstructions of character evolution are examined. MacClade's algorithms for implicitly or explicitly examining all MPRs are used to step through each unequivocal reconstruction of character evolution ([page 354](#)). In each unequivocal reconstruction, the number of i to j changes is counted. Some of these reconstructions may have more i to j changes than others. The minimum number of i to j changes among these reconstructions is recorded, as is the maximum number. When each character is examined (if "all included characters" is selected), the minimum and maximum numbers of i to j changes are added to running totals. The minimum number of i to j changes over all characters would be obtained if all of those reconstructions were chosen that minimize these changes; similarly for the maximum. In the bubble chart for the minimum and maximum option, there may be black spots contained within gray spots:



The black spot indicates the minimum frequency of changes (that is, the changes that occur on all reconstructions and all trees); the gray spot the maximum frequency (that is the changes that occur on some but not all reconstructions or trees).

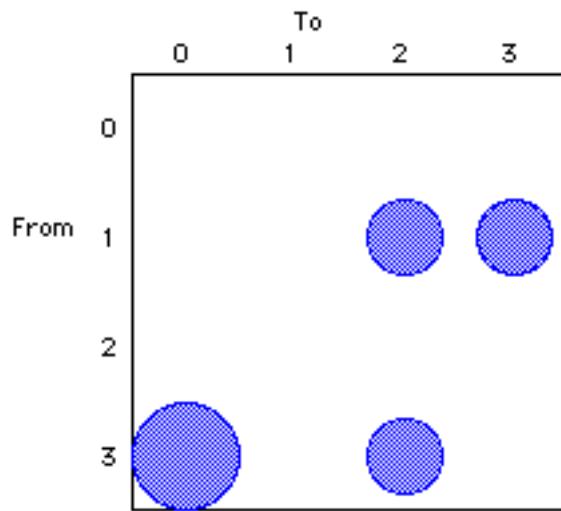
If several trees are examined, then for each tree there may be a minimum and maximum number of i to j changes, and over all trees there will be some tree that has the smallest minimum, and some other that has the largest maximum and average over all trees (see above, under "[Multiple trees" on page 383](#)). You can choose to display various sorts of maxima and minima: the average of the maxima over all the trees, the smallest minimum and largest maximum, and so on. (MacClade's default is to show the smallest minimum and largest maximum.) MacClade can show no more than three of these values at once (see "[Choosing the values to show" on page 406](#)).

When "average" is selected, all most-parsimonious reconstructions of character evolution are explicitly or implicitly examined. The number of i to j changes in each of these reconstructions is recorded, and MacClade calculates the average number among these reconstructions. That is, if there were four reconstructions and they showed 2, 2, 3 and 3 i to j changes respectively, then the average number of i to j changes is $(2+2+3+3)/4 = 2.5$.

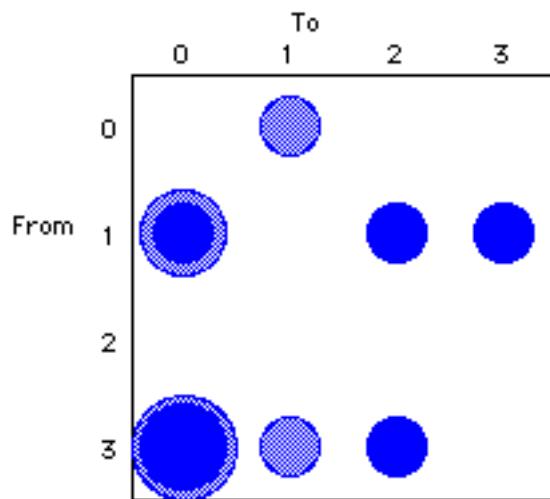
Because explicit or implicit examination of all MPRs is used for the minimum and maximum and the average options, these options cannot be used when certain factors, like the presence of polytomies, prohibit examination of all MPRs.



EXAMPLE: Open the "Bembidion" data file, and trace character 64 on the tree "1 of 4 common". Then ask MacClade to calculate a bubble chart, traced character only, showing only unambiguous changes. You should see a chart like this:

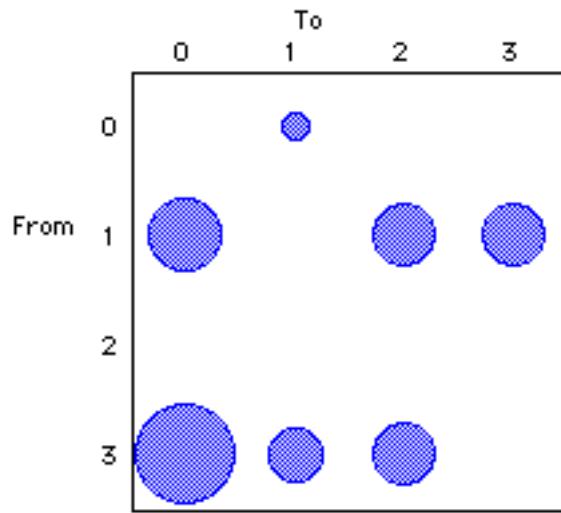


If you click on the largest circle, you will note that its area corresponds to 2 changes. If you then choose **Chart Options** from the **Chart** menu, and ask MacClade to calculate minimum and maximum changes, you should see a chart like this:



The largest gray circle here corresponds to 3 changes, as can be seen by choosing **Table** in the **View** submenu of the **Display** menu, or by touching on the circle. Note that there are some changes from 1 to 0 present in all MPRs but whose location is ambiguous, and thus these changes are not evident on the first chart, above. The fact that the gray and black spots in the minimum-maximum chart are not the same size shows that there is some ambiguity in the

reconstruction of ancestral states that affects the reconstructed number of 0 to 1 and 1 to 0 transformations. Finally, the chart of average changes is:



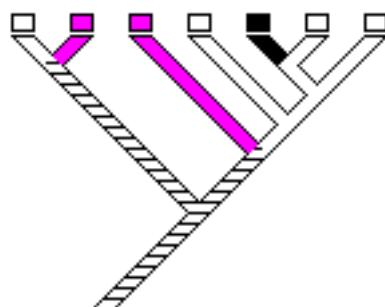
(The largest circle here corresponds to 2.6 changes.)

Trees examined

The State Changes & Stasis chart can be calculated on the current tree in the tree window or for multiple trees (either those stored in the current tree repository, another tree file, or random trees, the latter including multiple random resolutions of a polytomous tree on the screen). If the "stored trees" option is checked, MacClade will examine trees in the current external tree file, if there is one active; otherwise, it will examine trees stored with the data. If the "tree file" option is checked, MacClade will query you for the tree file you wish to examine.

Calculating total changes of a certain class

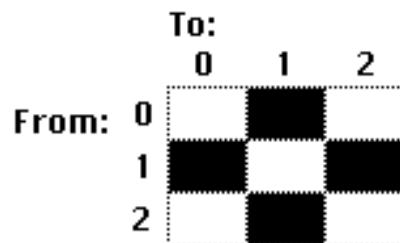
You can get a vague idea of the total number of changes of a certain sort by having a bubble chart displayed as a table (by choosing **Table** from the **View** submenu of the **Display** menu), and then summing the changes in the table. However, the sum can be misleading. For example, in the file "Total Changes Example", the single character in the matrix would be reconstructed like this:



The State Changes & Stasis chart of minimum and maximum changes will yield the following table:

| | | To: | 0 | 1 | 2 |
|--|--|-------|-----|-----|-----|
| | | From: | 0 | 0-2 | 1-1 |
| | | 0 | | | |
| | | 1 | 0-2 | | 0-0 |
| | | 2 | 0-0 | 0-0 | |

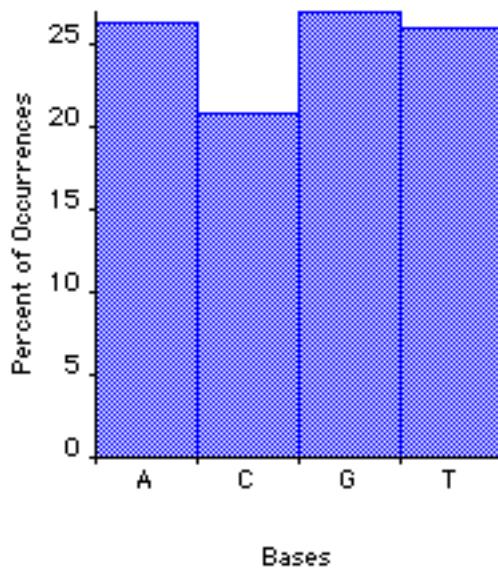
The sum of the minimum values in the To State 1 column, and the minimum values in the From State 1 column is 0; the sum of the maximum values is 4. This implies that there are between 0 and 4 changes to or from state 1, with variation due to the multiple equally parsimonious reconstructions of character evolution on the tree. But if you looked at each reconstruction of the character on the tree, it would be clear that each of them has exactly 2 changes to or from state 1. You can ask MacClade to calculate total changes of a particular class for you using the Total button in the **Chart Options** dialog box. The Total button will present you with a dialog box much like the **Restrict Changes** dialog box, described on [page 393](#). You then define a class of changes in the **Total** dialog box. In this example, you want to define the class of changes to be all changes to or from state 1:



Once this class of changes is set, then MacClade will total the number of changes of the defined class. To see the total, choose **Summary** from the **View** submenu of the **Display** menu. In this example, MacClade will calculate the number of changes to or from state 1 to be between 2 and 2.

States

This chart shows the frequency that various states occur among the terminal taxa of the current tree, as shown in the following example.



If no character is traced, the frequency in all (included) characters will be calculated. If a character is traced, you will be presented with a choice to calculate over the frequency in the character traced, or all (included) characters of the matrix. If the state of a taxon is uncertain, its possible states are not counted into the frequency. An example of use of a States chart is given in [Chapter 16](#).

If you ask to have the States chart displayed using percentages, then in the bar chart and table the values are shown with respect to the pool of observed states (not including missing data and gaps). Thus, if frequencies in the data matrix are 20% for each of A, C, G, T, and gaps, then the charts would show frequencies of A, C, G, and T of 25%. The Summary view calculates percentages in two ways: (1) as in the bar charts and tables; (2) with respect to the total number of data matrix entries (including missing data and gaps). The latter method would calculate percentages of A, C, G, and T as 20% for the example given.

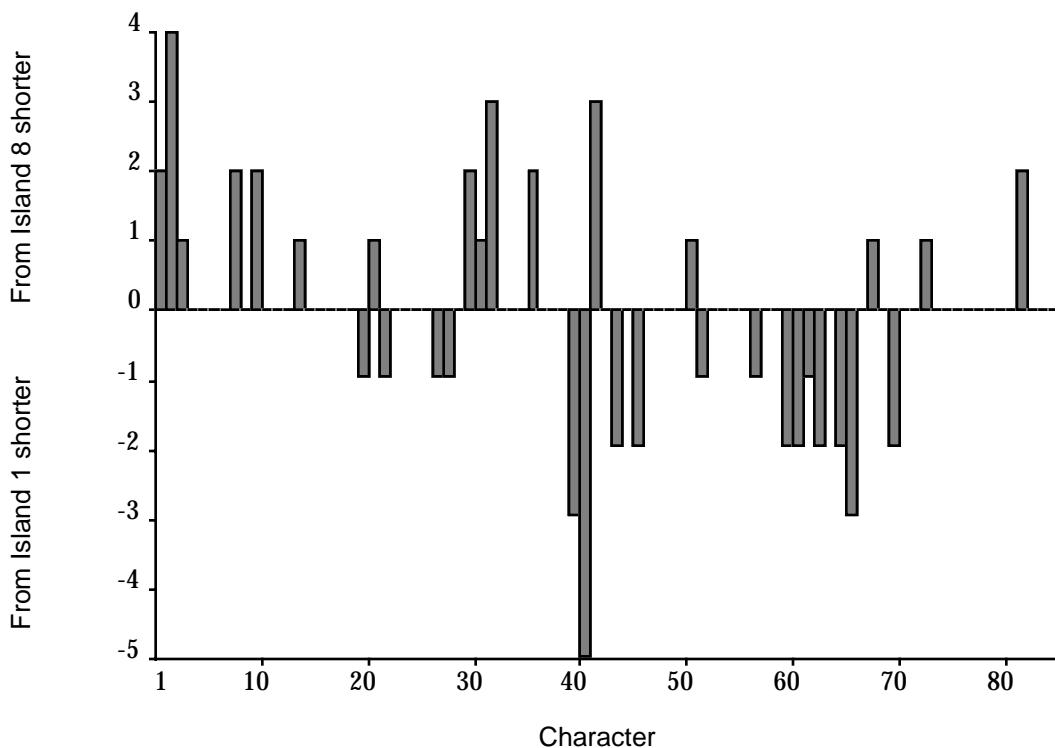
Comparing two trees

Choosing **Compare 2 Trees** from the **Chart** menu allows you to see quickly how two trees differ in the number of weighted steps assigned to each character. The two trees to compare are selected from the current tree repository (data file or tree file). In the first dialog that appears, select a stored tree. MacClade will beep; then you select another tree. MacClade will then present you with a graph of the difference in number of steps in each character between the first tree and second tree.

The Vertical Axis options dialog box (available in the **Display** menu) gives you the option to have the bars indicate either which characters are shorter in a tree or tree file, or which are longer.



EXAMPLE: Open the file *Bembidion*. Ask MacClade to Compare 2 Trees. In the dialog box that is presented, request automatic resolution of polytomies. In the next dialog box that is presented, choose tree "from Island 1"; in the following dialog, choose "from Island 8". You will be presented with the following chart:



The bars above the center line indicate that those characters exhibit less steps on the tree "from Island 8" than on the other tree; the bars below the center line indicate that those characters exhibit fewer steps on the tree "from Island 1". Note that character 41 is 5 steps shorter on tree "from Island 1" than on "from Island 8".

Comparing two tree files

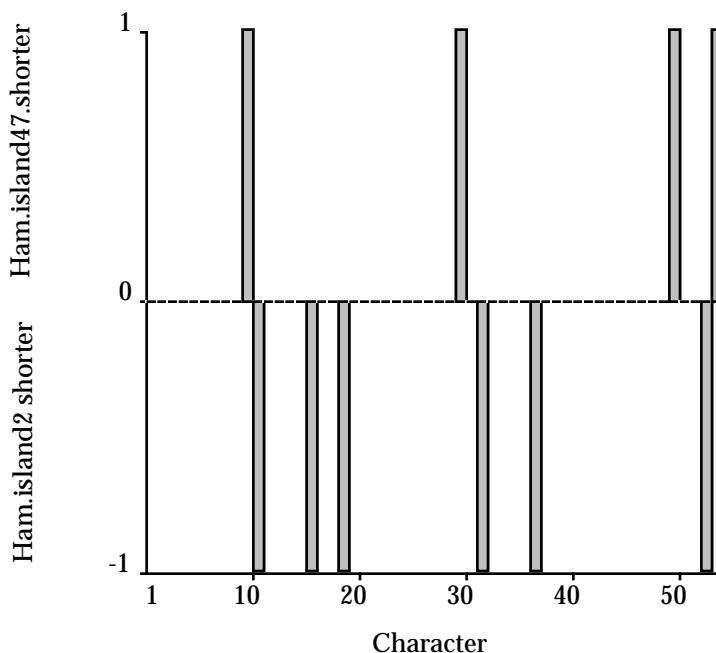
This chart displays, character by character, whether and by how much all the trees in one tree file differ from all the trees in another tree file. That is, it calculates whether the number of weighted steps required for a character by all of the trees in one tree file is consistently more or less than the number of steps required by all of the trees in another tree file.

This chart is requested by selecting **Compare 2 Tree Files** in the **Chart** menu. You will be presented with a dialog box to choose the first tree file, then a dialog box to choose the second tree file.

The Vertical Axis options dialog box (available in the **Display** menu) gives you the option to have the bars indicate either which characters are shorter in a tree or tree file, or which are longer.



EXAMPLE: Open the file "Hamamelidae". Go to the tree window. Ask MacClade to do a Compare 2 Tree Files chart. In the dialog box that is presented, request automatic resolution of polytomies. In the next dialog box that is presented, choose tree file Ham.island2 from the Hamamelidae Trees folder; in the following dialog, choose Ham.island47. These tree files contain two of the islands (D. Maddison, 1991a) of most parsimonious trees for this data matrix. You will be presented with the following chart:



Bars above the center line indicate those characters that are shorter on all Ham.island47 trees than all Ham.island2 trees; thus, there are four characters that are at least one step shorter on each Ham.island47 trees than on any Ham.island2 trees. Bars below the center line indicate characters that are shorter on Ham.island2 trees.

Chart display

Various options alter the manner of the presentation of the charts. Some of these options can be specified in the **Chart Options** dialog box that appears when you request the chart, as described above; others can be specified using the menu items in the **Display** menu (available when the chart window is the frontmost window).

Viewing charts, tables, or textual summaries

The items in the **View** submenu of the **Display** menu allow you to choose the basic form in which information is displayed: as a bar chart, table, bubble chart, or as a textual summary. All charts have at least one graphic view (bar chart or bubble chart) and one table view. The bubble chart is available only for the State Changes & Stasis chart. The textual summary presents information (such as number of characters and trees examined, and so on) not available in the other views.

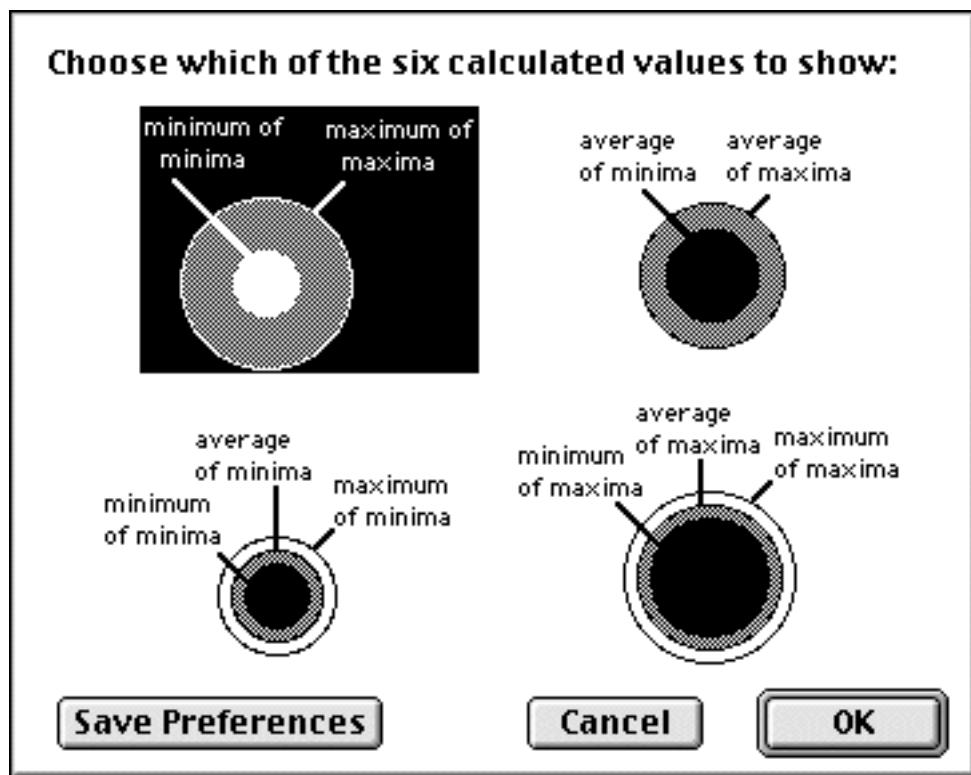
Choosing the values to show

Ambiguity in character reconstruction and multiple trees can yield more values for an interval of a bar chart or for changes from state i to j than can be displayed on one chart. In these instances there are six possible values that can be shown:

1. The minimum value across trees of the minimum value across reconstructions.
2. The average value across trees of the minimum value across reconstructions.
3. The maximum value across trees of the minimum value across reconstructions.
4. The minimum value across trees of the maximum value across reconstructions.

5. The average value across trees of the maximum value across reconstructions.
6. The minimum value across trees of the maximum value across reconstructions.

The **Values to Show** item in the **Display** menu brings forth a dialog box in which you can choose which two or three of these values are to be shown at once.



You have the choice to show:

- a. The two extreme values (the minimum of minima and maximum of maxima, values 1 and 6 above).
- b. The average values of the minima and maxima across reconstructions (values 2 and 5, above).
- c. The three values for the minima across reconstructions (values 1–3, above).
- d. The three values for the maxima across reconstructions (values 4–6, above).

Displaying a title

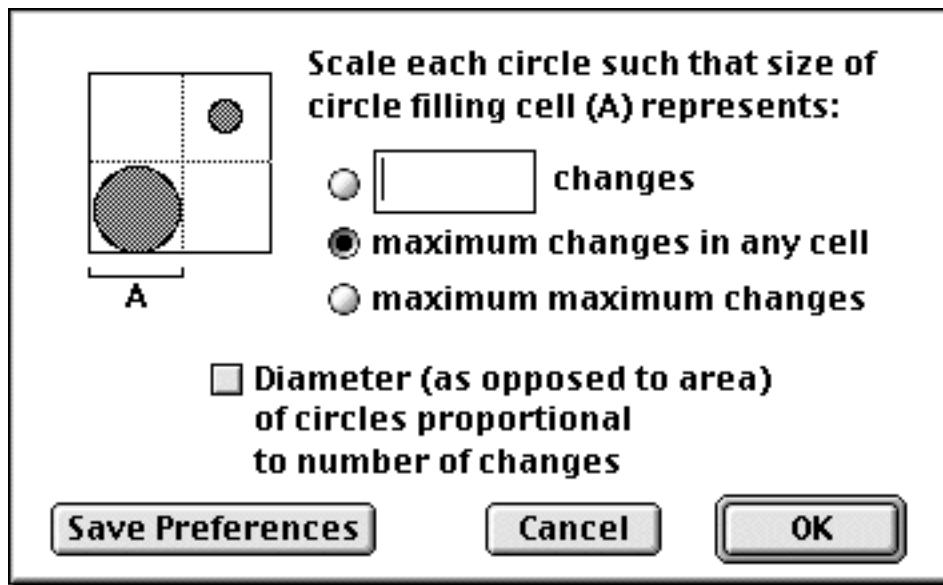
By default, MacClade includes a small descriptive title above the chart. To remove this, uncheck **Title** from the **Show** submenu of the **Display** menu.

Changing fonts

The **Font**, **Size**, and **Style** submenus of the **Display** menu allow you to choose the appearance of text used in the chart window.

Adjusting the size of circles

By default, MacClade adjusts the size of the circles such that the cell with the largest number of changes visible on the chart has a circle that exactly fills the cell, and scales the areas of the other circles in proportion to their values. You can alter MacClade's choice of circle sizes using the **Circle Size** item in the **Display** menu:



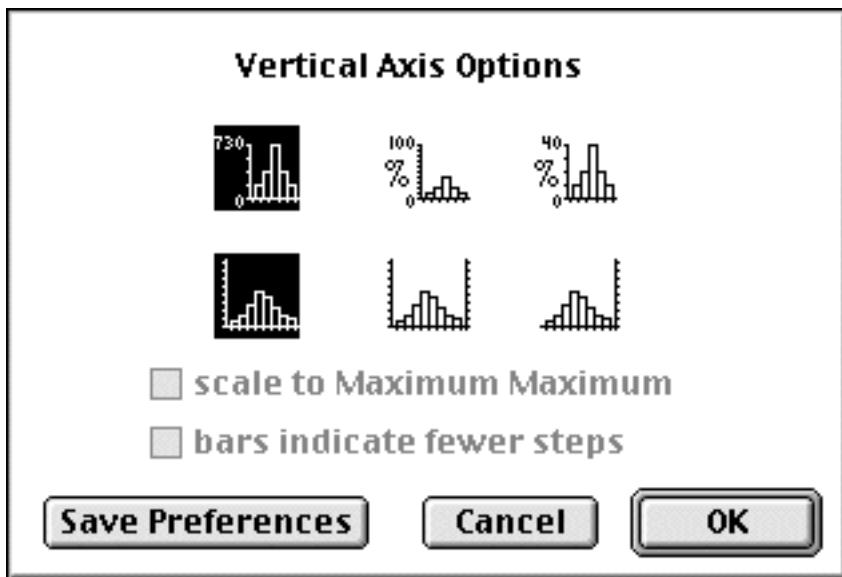
If multiple trees were examined, and you ask MacClade to scale the circles such that the size of each cell represents the maximum value over trees of the maximum number of changes over all reconstructions, then MacClade will shrink the circles so that even the maximum maximum number of changes would fit into the cells, even if this maximum maximum is not displayed on the current chart.

Either of these options may make it difficult to compare different charts, as the scales of the circles may vary from chart to chart. You can force a uniform scale across charts by manually specifying to scale the cell size to represent, say, 4 changes. MacClade will then adjust the sizes of the circles so that the width of the cells is equivalent to 4 changes. If there are more changes than this in some cells, the circles in those cells will be large, and will overlap adjacent cells.

By default, the area of each circle is proportional to the number of changes; this can be changed so that the diameter is proportional to the number of changes.

Adjusting the vertical axes

Options for vertical axes are available in the **Vertical Axis** dialog box, available in the **Display** menu:



The top three buttons determine whether the vertical axis shows (1) numbers or (2) percentages scaled from 0 to 100 or (3) percentages scaled from 0 to the highest observed. The bottom three buttons determine whether vertical axes are displayed on the left, both sides, or right.

If multiple trees were examined, and you ask MacClade to scale the vertical axis to accommodate the maximum value over all reconstructions over all trees, then MacClade will set the axis so that even the maximum maximum values would fit on the chart, even if this maximum maximum is not displayed on the current chart.

Adjusting the horizontal axis

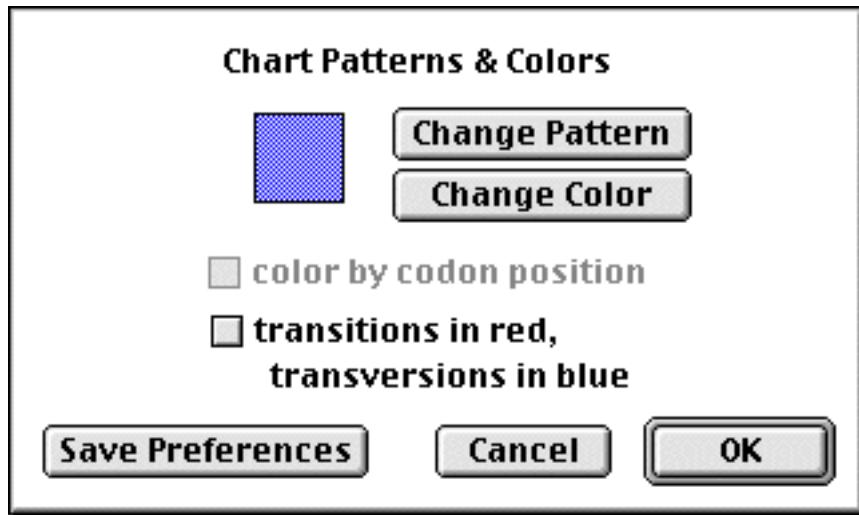
The **Horizontal Axis** item in the **Display** menu requests a dialog box in which the user chooses the horizontal scaling of the bars in a bar chart. MacClade automatically calculates a particular horizontal scaling, but the user can set a manual scaling if so desired. Width is measured in points (1/72 of an inch).

Changing display of State Changes & Stasis tables

For State Changes & Stasis tables, the entries in table view can be displayed either as absolute numbers, or as percentages. You can adjust this using the **Percentage & Norming** dialog box available in the **Display** menu.

Changing patterns and colors

Most patterns and colors used in charts can be changed using the **Patterns & Colors** dialog box, available in the **Display** menu:



You cannot change the patterns of the bars or bubbles if the chart is showing both minimum and maximum values.

In the State Changes & Stasis chart, for DNA or RNA sequence data, transitions will be shown in red, and transversions in blue if this option is selected. In the Character Steps/etc. chart, for DNA or RNA sequence data, if site number is shown on the horizontal axis, then you can ask MacClade to color the bars blue, green, or red for the first, second, and third codon positions respectively. Non-coding regions will be shown in black.

MacClade will allow you to change color even if the Macintosh you are using does not currently display color on the screen, if the machine is capable of displaying or printing colors (as is, for example, the Macintosh SE/30).

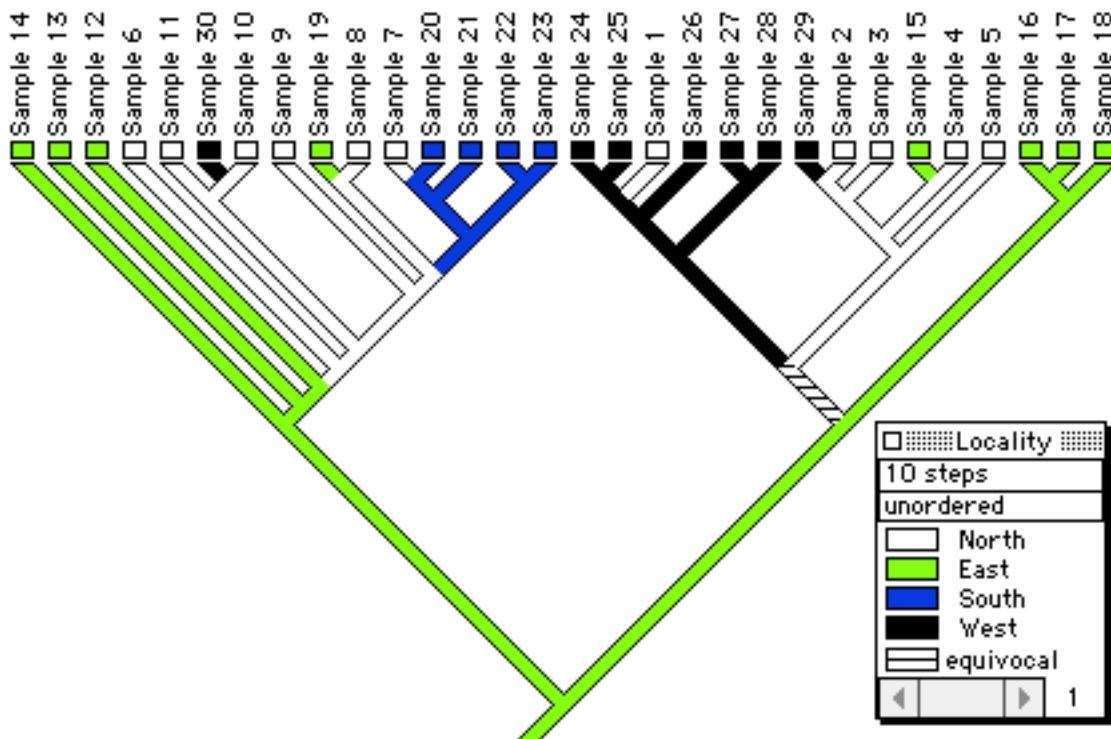
Displaying a grid

The **Grid** item of the **Show** submenu of the **Display** menu determines whether a grid is to be displayed on a chart.

Examples using molecular data

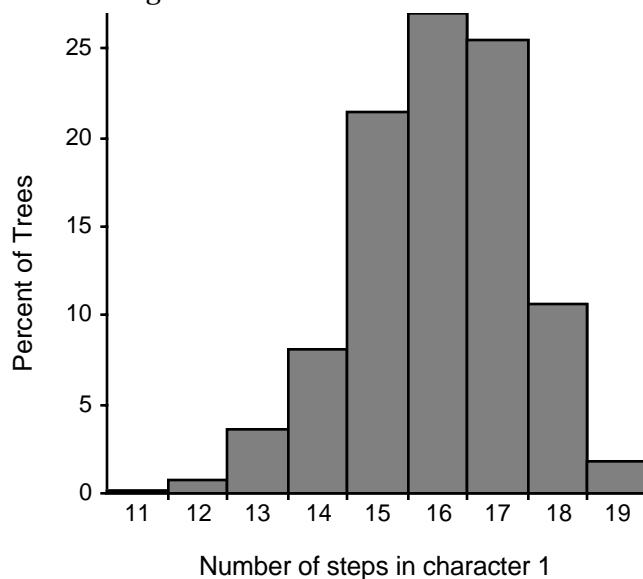
Testing panmixis using gene phylogenies

Suppose that you sampled individuals from four nearby localities and reconstructed a phylogenetic tree relating their mitochondrial DNA sequences, as follows:



How would you use this to test whether there is some restriction of gene flow among these populations? Slatkin and Maddison (1989) have suggested using the number of migration events required by a tree as a statistic to measure gene flow. Construct a MacClade data file listing all of the specimens and treat locality as a four-state unordered character. Trace this character to determine the minimum number of interlocality migration events allowed by the tree. In this case, it is 10. Then create a chart for the number of trees with

various numbers of steps for the traced character, using random trees. Construct 1,000 random joining trees. The chart should look something like this:



Were these four localities part of a single panmictic population, the minimum number of interlocality migration events should have the same probability distribution as the number of steps on random joining trees (Slatkin and Maddison, 1989; Maddison and Slatkin, 1991). Note that the 10 interlocality migrations on the reconstructed tree would be highly unlikely under panmixis, suggesting that there are indeed restrictions to gene flow.

Exploring transversions and transitions

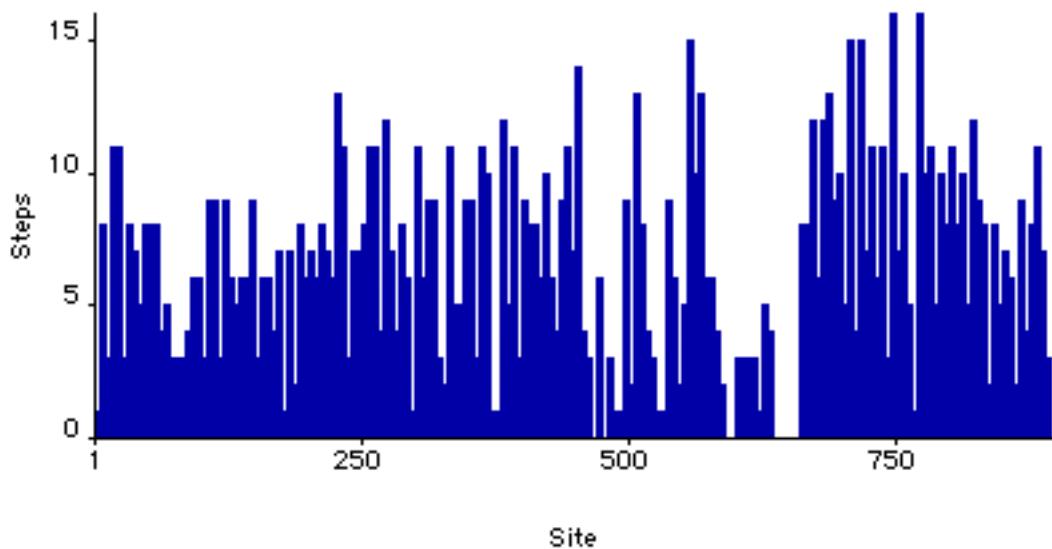


Open up the "Primate mtDNA" data file (from Hayasaka et al., 1988). It should open up to show the tree window. Choose **Character Steps/etc.** menu item of the **Chart** menu. Click the upper-middle chart icon



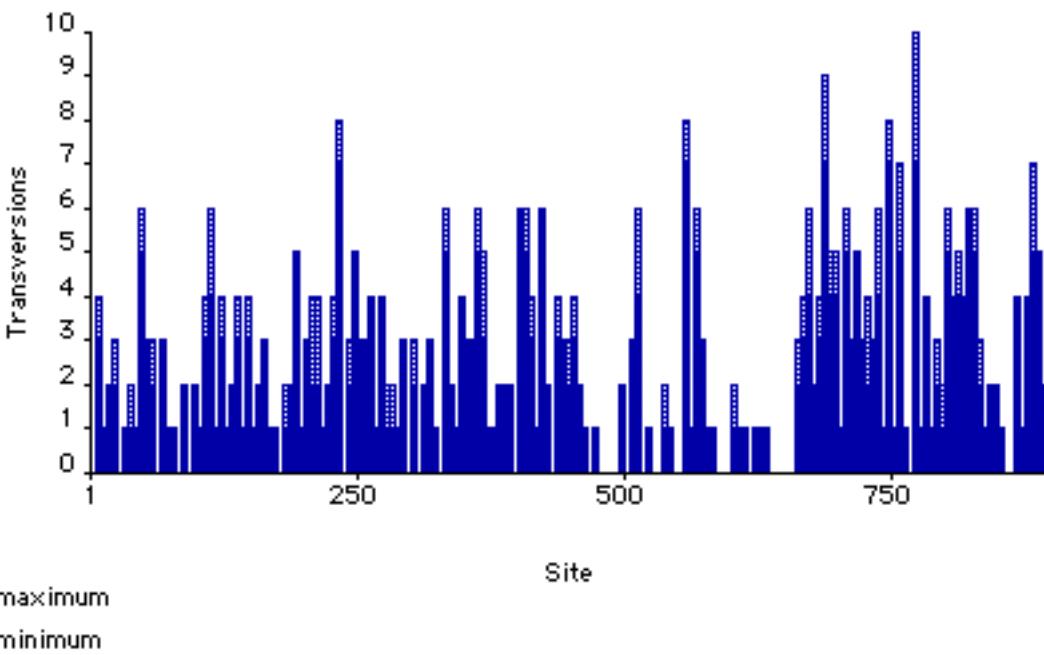
(charting by sequence for the current tree). Click on the button, and set the interval width to 5

using the dialog box that comes up; press OK in this dialog box. Then press Chart in the **Character Steps/ etc.** dialog box. You should see the following chart.

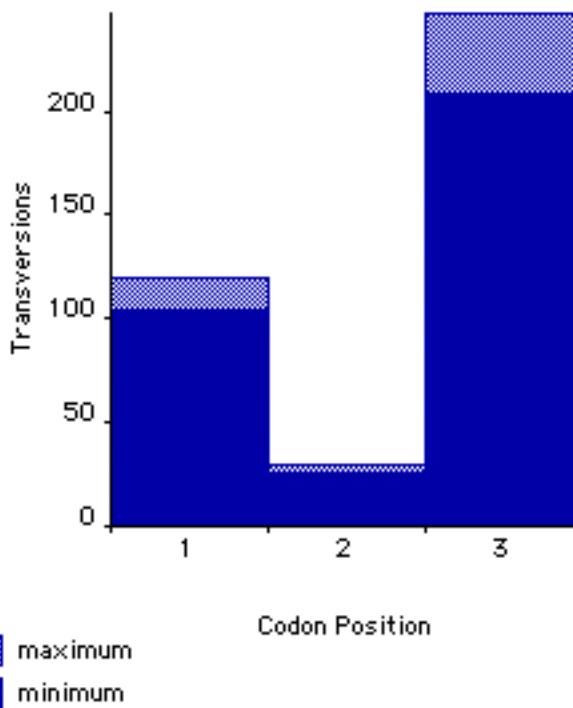


On the horizontal axis is the base position. On the vertical axis is the number of steps *on the tree* for each site in the DNA sequence. There is clearly a more conserved region around site 650. (This region codes for a leucine tRNA.)

If you choose **Chart Options** from the **Chart** menu, select the "changes" radio button on the right side, then click on Restrict, press the Transversions and OK button in the dialog box that appears, then press Chart. MacClade will show you the distribution of transversions along the sequence:

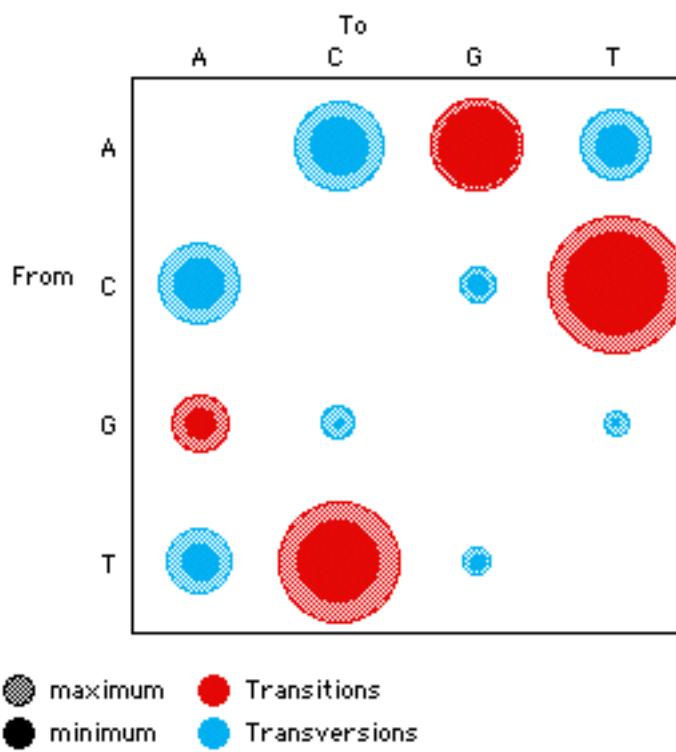


If you choose **Chart Options** from the **Chart** menu, and select the upper-right icon so that MacClade will chart transversions by codon position, the chart MacClade will present is:



As expected, most of the transversions on the tree are at the third position. Recall that these are changes reconstructed on the tree, not merely calculated from pairwise distances.

Choose **State Changes & Stasis** from the **Chart** menu. In the dialog box that appears, choose "count minimum and maximum number", and press Chart. You should see the following chart.



This chart is calculated over all sites, coding and non-coding, as no characters have been excluded. Note that the largest spots in each row are for the transitions A to G, G to A, C to T, and T to C. If you ask for a Table view, you will see these results in numerical rather than graphical form:

| | | To: | | | | |
|--|---|--------|---------|---------|---------|--|
| | | A | C | G | T | |
| | | From: | | | | |
| | A | | 72-179 | 136-186 | 41-116 | |
| | C | 56-149 | | 10-31 | 234-419 | |
| | G | 24-72 | 3-26 | | 1-14 | |
| | T | 30-100 | 147-334 | 5-18 | | |

The Summary view will appear as follows:

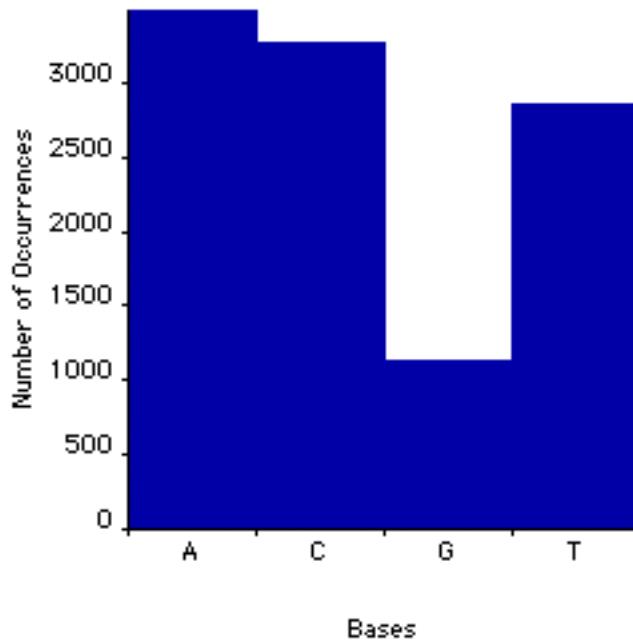
```
Number of transitions: 712-777  
Number of transversions: 376-441  
  
Number of changes in user-defined set "Transversions": 376-441
```

The number of reconstructed transitions is about twice as high as the number of transversions. Let us see if the results are similar when we focus only on second positions of codons. If you then choose the inclusion set "only 2nds" from the **Inclusion Set** list window (available from the **Inclusion Sets** submenu in the **Characters** menu), MacClade will present you with results for second positions only:

```
Number of transitions: 712-777  
Number of transversions: 376-441  
  
Number of changes in user-defined set "Transversions": 376-441
```

With second positions we can see that the bias for transitions is stronger, with about four times as many transitions as transversions reconstructed.

If you then choose the inclusion set **All Included** from the **Inclusion Sets** submenu from the **Characters** menu, and ask MacClade to show a **States** chart, you should see:



Note the low frequency of G.

In the Summary view, and you will get a summary of the frequency of each base:

Average and ranges of frequencies

across taxa in tree for states of all characters (missing data and gaps included):

| | | |
|---|-----------------|-----------------------|
| A | Average : 0.323 | Range : 0.304 - 0.347 |
| C | Average : 0.303 | Range : 0.253 - 0.344 |
| G | Average : 0.105 | Range : 0.098 - 0.116 |
| T | Average : 0.266 | Range : 0.236 - 0.290 |

Average and ranges of frequencies

across taxa in tree for states of all characters (missing data and gaps excluded):

| | | |
|---|-----------------|-----------------------|
| A | Average : 0.324 | Range : 0.305 - 0.349 |
| C | Average : 0.304 | Range : 0.254 - 0.345 |
| G | Average : 0.106 | Range : 0.098 - 0.116 |
| T | Average : 0.266 | Range : 0.237 - 0.291 |

The mean frequency of each base across taxa is listed in the first column, and the ranges of base frequency across taxa is listed after that.

Now let us consider alternative hypotheses about the closest relatives of *Homo sapiens*, using charts that compare hypothetical phylogenetic trees. Notice that the treelength of the tree in the tree window is 1153. Try rearranging the tree to give all three possible rearrangements of the *Homo-Pan-Gorilla* clade. Two of these, the one linking *Pan* with *Homo* and that linking *Pan* with *Gorilla*, both have a treelength of 1153, whereas the third arrangement linking *Gorilla* with *Homo* has a treelength of 1160. The first two equally parsimonious arrangements are stored in the data file as "*Homo-Pan*" and "*Pan-Gorilla*". To compare them, ask for the chart **Compare 2 Trees**. Choose one tree, then the other. You will notice that *Homo-Pan* requires one fewer step of each of 12 characters, whereas *Pan-Gorilla* requires one fewer step of a different 12 characters. By excluding first, second, or third codon positions you could examine whether one of the trees was supported by more first or second positions than the other. Now, go to the character list window, select all characters, and assign them all the user-defined step-matrix type "ttbias". As you can see in the **Type Edit** dialog box, this type assigns six steps to a transversion, and one step to a transition. When this assumption is applied to all characters, the treelengths of *Homo-Pan* and *Pan-Gorilla* are 3033 and 3048, respectively. As can be seen in the Compare 2 Trees chart, there are still 12 characters preferring *Homo-Pan* and 12 preferring *Pan-Gorilla*, except that in three of these characters, *Pan-Gorilla* requires a transversion. The chart shows these characters at positions 88, 340, and 625. In the character list window the first two can be seen to be third positions, the last in a non-coding region. If a step matrix is used that gives a higher cost to transversions than transitions, then *Homo-Pan* is preferred.

Confirming the validity of the charting calculations

In programming MacClade, the proper functioning of the charting calculations was tested as follows:

- The source code was thoroughly checked and rechecked.
- For several test cases, the results of the charts were compared against results calculated by hand, and as calculated by other parts of MacClade (e.g., the results presented in the character list window).



21

CONTINUOUS CHARACTERS

The preceding several chapters dealt with characters with a finite number of discrete states (0, 1, 2, and so on). MacClade can deal directly with continuous-valued character data, though its features are relatively limited.

To make discrete or not?

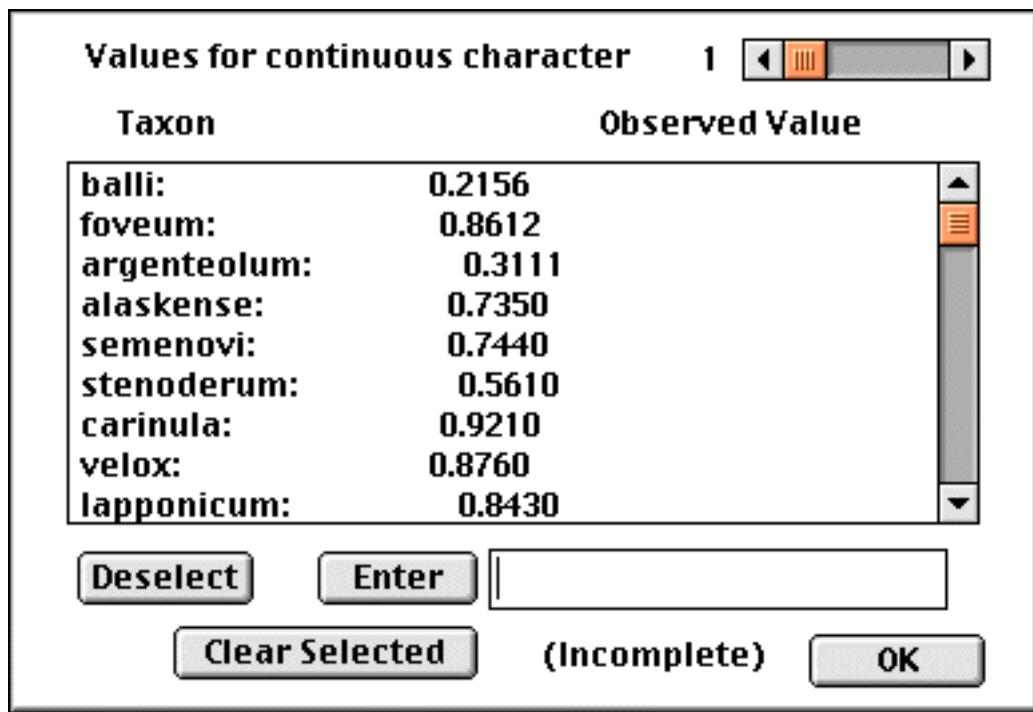
Continuous characters have often been recoded as discrete characters for use in phylogenetic analysis (Archie, 1985; Mickevich and Johnson, 1976; Felsenstein, 1988b; Baum, 1988; Goldman, 1988; Chappill, 1989; Stevens, 1991), though at least for the purpose of reconstructing character evolution there exist methods that can handle continuous characters without recoding (Farris, 1970; Rogers, 1984; Swofford and Berlocher, 1987; Swofford and Maddison, 1987; Huey and Bennett, 1987; W. Maddison, 1991). MacClade allows both options, recoding to discrete or treating continuous characters in their native form.

Continuous characters can be recoded as discrete and entered into MacClade's data editor, and treated as ordered characters. If each distinct observed value of the continuous character receives a different discrete state in the recoding, and the order of values is preserved in the order of states 0, 1, 2, ..., then the reconstruction of ancestral states will not be affected by the recoding, at least if you are using Wagner or linear parsimony (Maddison and Slatkin, 1990). That is, the reconstruction of character evolution obtained with the discretely recoded character treated as an ordered character would be the same as the reconstruction obtained using the Wagner/linear parsimony option for continuous characters (see below). Discrete recoding in MacClade allows you to take advantage of several features such as charting, explicit or implicit examination of all MPRs, fixing of states at branches, and soft polytomies that are available only with discrete characters, but it has several disadvantages. First, the recoding has to be done. Second, squared-change parsimony cannot be used on discrete characters. Third, although the reconstruction of character evolution may not be changed by recoding, the character's length required by the tree may change because the distance between two adjacent states in an ordered discrete character is counted as one step, whereas values in a continuous character may be separated by varying distances. (This last problem may be solved if the discrete character is assigned a user-defined transformation type with state-to-state distances reflecting the distances in the original continuous form.) Fourth, MacClade can handle only 26 discrete states in a character. If a continuous character has more than 26 different values in a set of taxa, then it cannot be recoded to discrete without loss of information.

MacClade's features for treating continuous characters in their native form are described below.

Editing continuous characters

Select **Edit Continuous** in the **Edit** menu to obtain a dialog box listing the taxa in the data matrix and their assigned values in the continuous characters. At the top is a scroll bar to choose which of the 10 continuous characters is being edited.



The Edit Continuous dialog box

Select one or more taxa whose values you wish to change (using the standard methods for selecting elements in a list), type in the new value in the box below the list, and click on the Enter button. You can type in a range (e.g., "0.89 – 0.94"), though only the lowest value will be used when squared-change parsimony is in effect. To expedite entry of data for many taxa, you can hit Tab to simultaneously enter the value you have typed and move to the next taxon. Click on OK when you are done editing.

The **Edit Continuous** dialog box allows you to indicate a range for a single terminal taxon by using "to" or "-", as in "0.13 to 0.25" or "2.31 - 3.44". You can also indicate small or large numbers using exponential notation, as in "3.5E3" (for 3500.0) or "2.8E-6" (for 0.0000028). However, in the same entry you cannot use both negative exponents and "-" to indicate a range. Thus "1.4E-6 - 2.8E-6" is not allowed and will not be interpreted correctly. Continuous values can be from 0.0 to 9999.9. Larger values and negative values are not allowed.

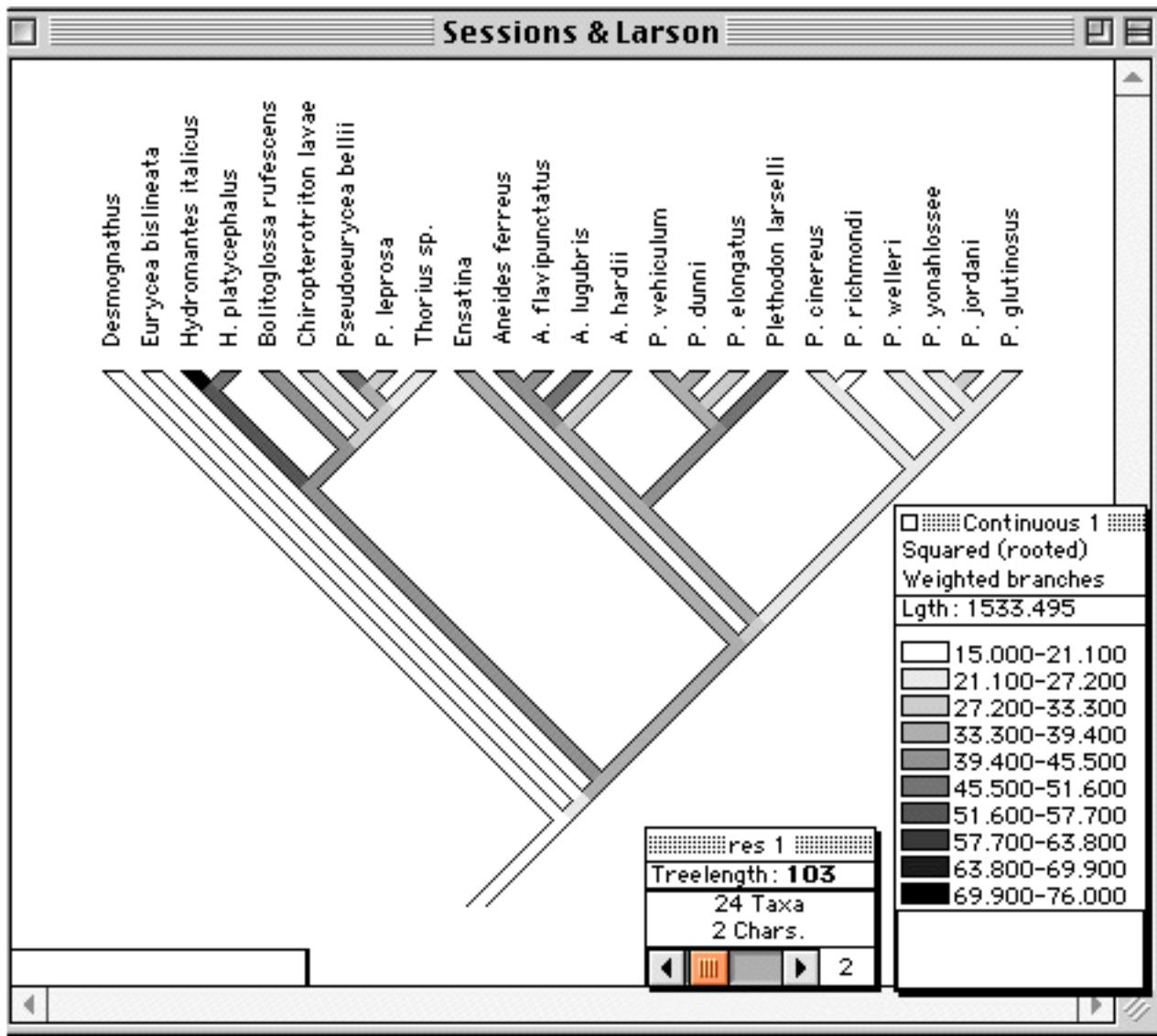
No missing data are allowed when a continuous character is traced on the tree. Thus, MacClade will indicate an error if you try to trace a continuous character on the tree and it finds a missing value. In the **Edit Continuous** dialog box, missing values are indicated by blank spaces beside the taxon names. MacClade indicates "Complete" at the bottom when there are no missing values in the character shown.

The Deselect button changes all taxa in the list from selected to unselected. The Clear Selected button sets all selected taxa to have no specified value for the character.

Continuous characters are saved in a special block of the data file, resembling the data matrix for the discrete characters. Even if you have only continuous characters you must still work with the discrete-character data editor, if only to create the taxa and their names.

Tracing continuous characters

MacClade will reconstruct the evolution of a continuous character on the current tree in the tree window, as shown in the example, below:

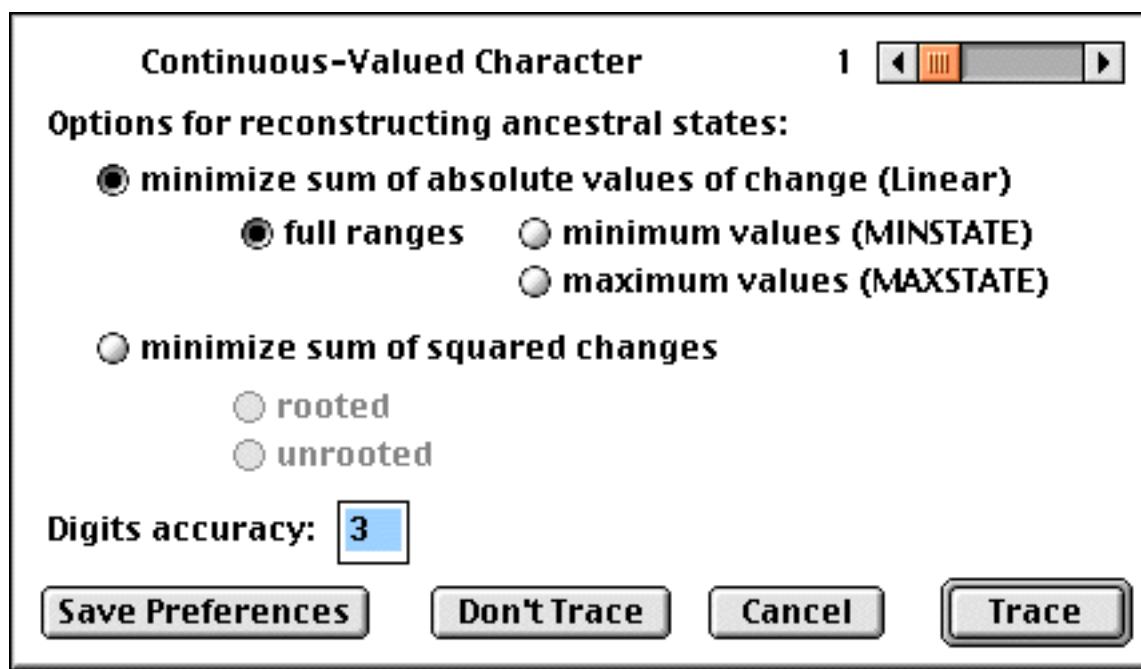


The tracing of continuous characters can use two different parsimony criteria to reconstruct ancestral states:

1. "Linear", "Manhattan", or "Wagner parsimony" minimizes the sum of the (absolute value of the) changes on the branches of the tree (Farris, 1970; Swofford and Maddison, 1987). This often yields a range of equally parsimonious values at each branch. MacClade will display the full range, or only the minimum (MINSTATE) or maximum (MAXSTATE) values at each branch according to choices made in the **Trace Continuous** dialog box. Linear parsimony can be used only with dichotomous trees. MacClade uses Swofford and Maddison's (1987) algorithm for linear parsimony. See "[ACCTRAN and DELTRAN](#)" on page 98 for a discussion of MINSTATE and MAXSTATE.

2. "Squared-change parsimony" minimizes the sum of the squared changes on the branches of the tree (Rogers, 1984; Huey and Bennett 1987). This yields a single value at each branch. With squared-change parsimony, terminal taxa are not allowed to be polymorphic. If a range is indicated, only the minimum value will be used. Whether or not the root of the tree is considered a true node in the tree will affect the reconstruction (W. Maddison, 1991). Thus, rooted or unrooted options can be chosen. If the tree has a basal polytomy, then the unrooted option cannot be chosen. Squared-change parsimony can be used with dichotomous trees and trees with hard polytomies. MacClade uses W. Maddison's (1991) algorithm for squared-change parsimony. (See McArdle and Rodrigo, 1994, for an alternative formulation using the algorithms of matrix manipulation instead of those of tree recursion.) As noted by W. Maddison (1991), the squared-change reconstruction can also be considered a Bayesian probability estimate under a Brownian motion model of evolution (see also Schlüter et al., 1997).

These options are specified in the **Trace Continuous** dialog box, available in the **Trace** menu:



The **Trace Continuous** dialog box

If a continuous-valued character has been read or entered, its evolution can be reconstructed by requesting this dialog box. Choose continuous character 1, 2, 3, or 4 and click on **Trace**. If a taxon in the tree has not been assigned a continuous value, a warning is given and the continuous tracing is turned off. To turn off the trace, call up the dialog box and click on **Don't Trace** or, alternatively, click on the go-away box in the upper left of the continuous character legend. To trace a different continuous character, either go back to the **Trace Continuous** dialog box and choose a different character, or click on the left and right arrow keys to scroll from one continuous character to the next.

The tracing will automatically break the observed range of the character into 10 subranges and assign each a color, shade, or pattern, indicated in the legend at right. If a node is reconstructed to have a range of values spanning two or more subranges, it is shaded by the equivocal pattern. When the cursor is held over a branch, the exact value or range of values reconstructed for that branch is shown in the character legend. Using the **Trace Labeling** dialog box in the **Display** menu, you can ask MacClade to place labels on each branch to indicate textually the states reconstructed.

The continuous character tracing can be printed using **Print Tree**, and a listing node-by-node of the reconstructed ancestral states can be printed by selecting **Branch List** in the **Print Other** submenu, or saved to a text file using the **Branch List** item in the **Save Text File** submenu.

Confirming the validity of the parsimony algorithms

In programming MacClade, the proper functioning of the continuous-character parsimony algorithms was tested as follows.

- The source code was thoroughly checked and rechecked.
- Ancestral state reconstructions and counts of treelength were examined by eye in a number of test characters.
- Using the principle described by Maddison and Slatkin (1990), the discrete-character algorithms were used as a check against the linear parsimony algorithms for continuous characters.
- The results of the squared-change parsimony algorithms were checked against the results from an alternative algorithm, the iterative averaging algorithm (Rogers, 1984; Huey and Bennett, 1987), for a number of test cases.



PATTERNS OF CORRELATED CHARACTER EVOLUTION

22

Among the various applications of phylogenies discussed in the literature (e.g., Maddison and Maddison 1992: Chapter 3), one that has attracted much attention recently is the use of phylogenies to uncover patterns of correlated character evolution. MacClade provides a number of features to aid in applying tests of correlated character evolution.

The one correlation test that is directly implemented in MacClade is W. Maddison's (1990) concentrated-changes test. It can test whether changes in one binary character are more concentrated than expected by chance in certain regions of the phylogeny.

Using reconstructed histories to test correlation

One function of MacClade that is relevant in seeking patterns of character evolution is its tracing of character evolution. A number of proposed tests of character correlation involve first reconstructing the ancestral states in the characters of interest (Ridley, 1983; Huey and Bennett, 1987; W. Maddison, 1990; Sillén-Tullberg, 1993). Both for discrete and continuous characters MacClade supplies parsimony reconstructions of ancestral states. When the exact reconstruction is equivocal, MacClade's various features for examining alternative reconstructions can be used. Show MPRs mode for discrete characters allows you to examine all most-parsimonious reconstructions of ancestral states. This can be important, for example, if some reconstructions would be favorable to the correlation hypotheses, and others not.

Applying correlation tests to reconstructions of character evolution is a practice that can pose some problems. The reader is advised to see the comments in "[Should we rely on historical reconstructions?" on page 62](#)" regarding reliance upon ancestral state reconstructions.

The concentrated-changes test

The concentrated-changes test of W. Maddison (1990) is designed for testing the association of changes in a binary character with some other binary variable within a selected clade. For instance, it can test whether gains (0 to 1 changes) in one character are more concentrated than expected by chance on those branches of the clade selected that are reconstructed to have a particular state in a second variable. A significant concentration might indicate that this state in the second character may enable or select for gains in the first character. Note that in MacClade's implementation of the test, a 0 to 1 change is called a gain; a 1 to 0 change a loss. If you want to assume the reverse, recode the character ("[The Recode dialog box" on page 238](#)).

Calculations are requested by touching the tool on the branch below the clade over which the test will be performed. Before requesting the calculations, there must be a binary character traced on the tree to serve as the independent variable.

The calculations determine the probability that various numbers of gains and losses would occur in certain distinguished areas of the clade selected, given a certain number of gains and losses occur in the whole clade, and given the null model that changes are randomly distributed among the branches of the clade. One can imagine the test scattering changes randomly around the clade many times, and counting the

fraction of replicates in which certain numbers of gains and losses are located in the distinguished area of the clade. If the null model suggests that the number of gains or losses actually reconstructed in this area should only rarely occur by chance, then a large number reconstructed in this area would suggest the null model is false. That is, changes are not distributed randomly, and they are significantly concentrated either within or outside of the distinguished area. For more information see W. Maddison (1990) and Harvey and Pagel (1991:88).

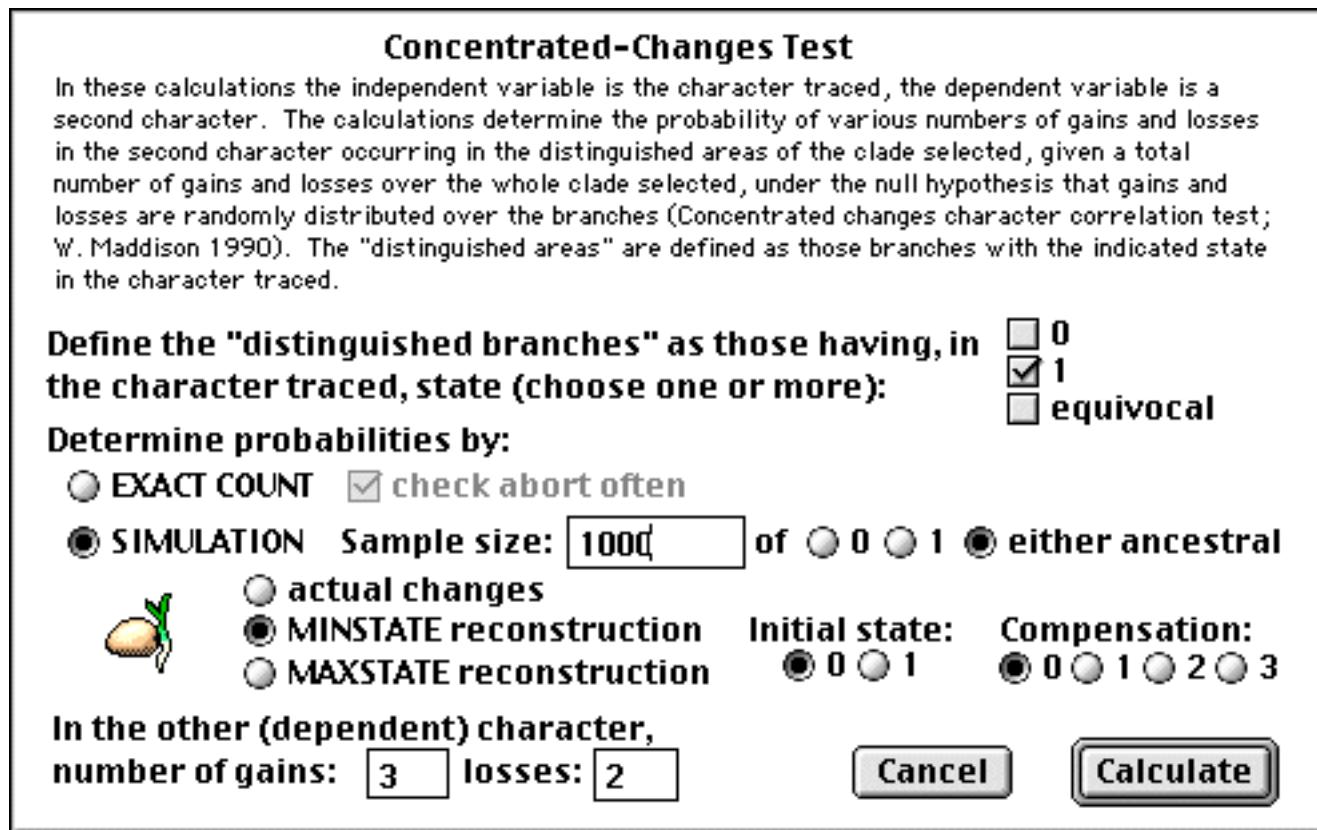
To perform the test, trace the evolution of a binary (two-state, 0–1) character representing the independent variable. MacClade uses this character's tracing to decide what you intend to be the distinguished branches. That is, the tracing in this character will indicate those branches on which a significant concentration of the other (dependent variable) character is sought. By choosing 0, 1, or equivocal in the **Correlation Test Parameters** dialog box (see below), you indicate whether the distinguished branches are those having state 0, 1, and/or equivocal in the character traced.

Thus, if the hypothesis tested is that proximal chiasma localization is a constraint against the evolution of chromosome fusions, chiasma localization (distal versus proximal) is traced on the clade. Use proximal=state 0 and distal=state 1, and indicate that the distinguished branches are those with state 1. You would then ask MacClade to calculate the null distribution for the number of gains of fusions falling in the distinguished area. If gains appear more concentrated than by chance, then there would appear to be an association between fusion and the supposedly unconstrained condition.

If needed, you can use the paintbrush tool to fix the state at various branches, for instance to resolve ambiguity. You can even create a fictitious character and trace it purely to define distinguished branches. For instance, if you want to ask whether changes in the dependent character are concentrated on branches on which the independent character changes, you could invent a character with all state 0s, then fix to state 1 just those branches on which the independent character changes, and set 1 to be the distinguishing state. Remember that the character traced must be the independent variable, and be a binary character (states 0 and 1 only).

NOTE: *The concentrated-changes test can be applied only to dichotomous trees. If your tree is polytomous, then you will have to resolve it manually or examine one or more random resolutions. Trees also cannot have any observed taxa fixed as ancestors.*

Once you have prepared the character tracing as you wish, the character-correlation tool () is then applied to the base of the clade over which the calculations are to be done. MacClade will present the following dialog box, in which you specify options for the calculations.



Correlation Test Parameters dialog box

The next several sections outline how you set up the test using this dialog box.

Choosing the branches distinguished by the independent character

With the choice of "distinguished branches" you indicate the character state(s) in the traced character indicating the area of the clade in which a concentration is sought. Thus, if equivocal and 1 were checked, the concentration test would determine the probabilities of various numbers of gains and losses falling on those branches of the clade reconstructed as have state 1 or an equivocal (0,1) assignment.

Methods to calculate probabilities

Two methods can be used to determine probabilities. The exact count uses the formulae presented by W. Maddison (1990). If the problem is too large for exact calculations, simulations can be used.

Exact count

In order to make the exact count calculations reasonably fast, four-byte integers are used instead of real numbers. Because the calculations can yield very large numbers if there are many observed changes in the dependent character or taxa in the clade selected, and because four-byte integers can be only as high as 2,147,483,647, the calculations may exceed this value and therefore cannot proceed further. Approximate overflow conditions are as follows: with 130 taxa, the highest number of gains or losses in the dependent

character that does not overflow is 2; with 80 taxa, 3; with 40 taxa, 4; with 30 taxa, 5; with 20 taxa, at least 6. If overflow occurs, MacClade will stop the calculations and give a warning. The exact calculations, if there are more than 5 or 6 observed gains or losses in the clade, can be very slow.

Simulations

If the exact calculations overflow or are too slow, simulations are a reasonable alternative. The simulations generate changes randomly on the clade selected and the number of gains and losses within the area of interest and in the clade as a whole are counted. Only those replicates that match the observed total number of gains and losses (entered at the bottom of the **Correlation Test Parameters** dialog box) are examined to see where the gains and losses are distributed. The simulation continues until the requested sample size is reached, or the user hits Command-period ($\text{⌘} -$).

In the simulations, MacClade first decides how many changes it will throw onto the clade. If the simulation is of actual changes, then it will determine number of changes to be the sum of the "observed" total number of gains and losses indicated at lower left in the **Correlation Test Parameters** dialog box. Then, that many branches are randomly chosen using uniform random numbers. *Note that the root branch of the clade selected is never chosen for a change.* MacClade then supposes 0 is ancestral, and moves up the tree interpreting the changes as either gains or losses. If the total number of gains and losses each match the requested "observed" number, then MacClade accepts this as an instance of the requested number of gains and losses, given 0 is ancestral. If the simulated number of gains matches the requested number of losses, and vice versa, then MacClade accepts this as an instance of the requested number of gains and losses, given 1 is ancestral. In either case MacClade counts how many of the gains and how many of the losses are in the distinguished areas of the clade selected. One is added to the appropriate cell in a matrix storing the observed frequency of each possible combination of gains and losses in the distinguished areas. At the end of the simulations the frequencies in this matrix, divided by the sample sizes simulated, are taken as estimates of the probabilities of various numbers of gains and losses in the distinguished areas.

If the simulations are of reconstructed changes, then MacClade will throw onto the clade a number of changes equal to the requested "observed" number of gains plus losses plus the compensation (see below). Assuming the indicated initial state at the root of the clade selected (see below), these changes imply states in the terminal taxa. Using these terminal states, the ancestral states in the clade selected are reconstructed using the typical parsimony algorithms. Ambiguity in these states is resolved using either MINSTATE or MAXSTATE; this is done instead of ACCTRAN/DELTRAN, which can leave ambiguities unresolved (see [page 98](#)). If the reconstructed number of gains and losses matches the requested number of gains and losses, then MacClade accepts this as an instance of the requested number of gains and losses given the reconstructed ancestral state is ancestral; the calculations proceed as for the actual changes. Note that the reconstructed ancestral state in the simulation is based entirely on simulated states at the terminal taxa in the clade selected; it is assumed that no information is available from outside the clade.

Option for simulations

Sample size

In case you intend to focus on cases with a particular ancestral state in the dependent character, MacClade allows you to qualify the sample size requested. Thus you may ask to simulate 1,000 replicates in which state 0 is ancestral. Along the way, other replicates will be found with 1 ancestral, but the simulation will stop when 1,000 replicates with 0 ancestral have been found. The maximum sample size is 1,000,000.

Actual vs. MINSTATE/MAXSTATE

With simulations, there are three options: the actual changes can be examined, or the changes as reconstructed by parsimony using MINSTATE or MAXSTATE ambiguity reduction (see "[ACCTRAN and DELTRAN](#)" on [page 98](#)) can be examined. By using reconstructed changes, you are using the null model as stated (equal probability of change on each branch), but in deriving the probability distributions, you are

accounting for possible biasing effects of the use of parsimony reconstructions (W. Maddison, 1990:553). With the reconstructed change options you are asking the question "What are the probabilities of reconstructing changes in various places given the null model?" rather than "What are the probabilities of changes actually occurring in various places given the null model?". The former might be considered in general better, because in practice we only have reconstructions, and are never certain we know the actual changes. However, the MINSTATE and MAXSTATE options present two extra parameters to specify, initial state and compensation.

Initial state

For simulation of reconstructed changes (MINSTATE or MAXSTATE) you need to assume an initial state (at the root of the clade selected) in the simulation so that MacClade can interpret whether changes are gains or losses, and so assign states to terminal taxa from which to reconstruct changes.

In the exact count, you do not need to prespecify an ancestral state. By assigning certain numbers of gains and losses randomly to the branches, as a side consequence you end up implying an ancestral state at the root node of the clade selected. This state will be sometimes 0, sometimes 1. It should be noted that (a) the probability of 0 or 1 being implicitly ancestral depends on the shape of the tree and other factors, and (b) the probability distribution of numbers of gains and losses falling in the distinguished areas conditional upon 0 ancestral may be different from that conditional on 1 ancestral. Recall that when we say 1 ancestral, we mean at the root node of the clade selected, that is, at its "ingroup node" (Maddison et al., 1984), not the next node lower.

In the simulations, the initial state need not be specified when actual changes are counted but it needs to be specified when reconstructed changes are counted. With actual changes, it is not clear whether a change is a gain or a loss until one moves up the tree with an ancestral state assumed. However, suppose that one assigned changes to branches, and then tried both 0 and 1 ancestors. If the number of gains and losses desired were different, and the appropriate number of total gains and losses (requested by the user) occurred assuming 0 was ancestral, then this could be counted as an instance of the correct numbers and ancestor. This is what MacClade does. It can be shown that the various probabilities, including that of choosing ancestor 0 versus 1, match those from the exact calculations.

After simulating reconstructed changes you will note that you can still ask a question such as "Given that we have reconstructed 4 gains 2 losses and 0 at the base, what is the probability of so many gains in the distinguished areas?" even though in the **Correlation Test Parameters** dialog box you specified 1 as the initial state. The distinction here arises because you might reconstruct state 0 as ancestral even if 1 actually was ancestral. However, you should realize that the assumed simulation ancestral state is not necessarily the reconstructed one, and you may get different probabilities asking the above question depending upon whether you had set the initial state to 0 or 1.

Compensation

When you are simulating reconstructed changes with the MINSTATE or MAXSTATE options, compensation needs to be specified. Compensation is a variable indicating how many extra changes are desired. It is needed because parsimony reduces the number of changes. If you need to generate 6 changes on the clade selected, then by having a compensation of 1 or 2, one would throw down 7 or 8 actual changes, and this would yield a better chance of parsimony reconstructing 6 than if you had thrown down only 6 actual changes. Note that this is not just an issue of speed; it could affect probabilities. Generally, you can leave the compensation at 0 unless the total number of changes is more than one-quarter of the number of taxa in the clade. MacClade monitors the simulation periodically to check if changing compensation would appear to speed the simulation; if so, the user is given the option of stopping the current simulation.

Setting changes in the dependent character

At the bottom of the dialog box, you enter the number of gains and losses reconstructed in the character

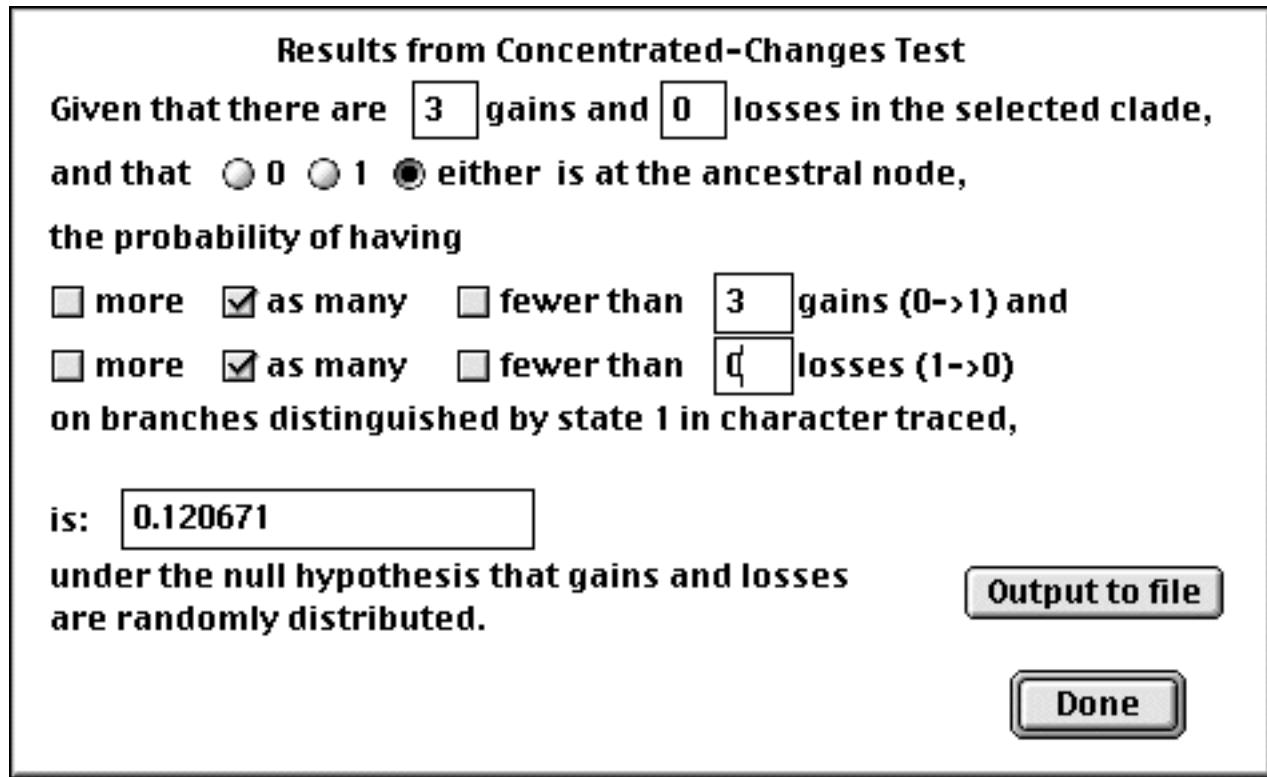
whose concentrated evolution is being examined. This is the number of gains and losses in the whole clade selected. The calculations will then derive the probabilities of various numbers of gains and losses in the area of interest, given the indicated number of gains and losses in the whole clade selected. Note that even if you put 3 gains and 1 loss in this dialog box, after the calculations are done you can ask about probabilities given there are *other* numbers of gains and losses in the whole clade selected (up to 3 gains and 3 losses) as long as the calculations are done by exact count. If the calculations are done by simulation, you can ask only about the same total number of gains and losses indicated initially.

Stopping calculations

The calculations can be stopped by hitting Command-period (⌘-.). MacClade checks fairly frequently to see if you have requested a stop, but this checking slows down the calculations. To speed up the exact calculations, but therefore check less frequently, turn off "check abort often".

Viewing probabilities

After the calculations, a dialog box will be presented that allows you to ask about the results, as shown in the following figure.



Correlation Test Results dialog box

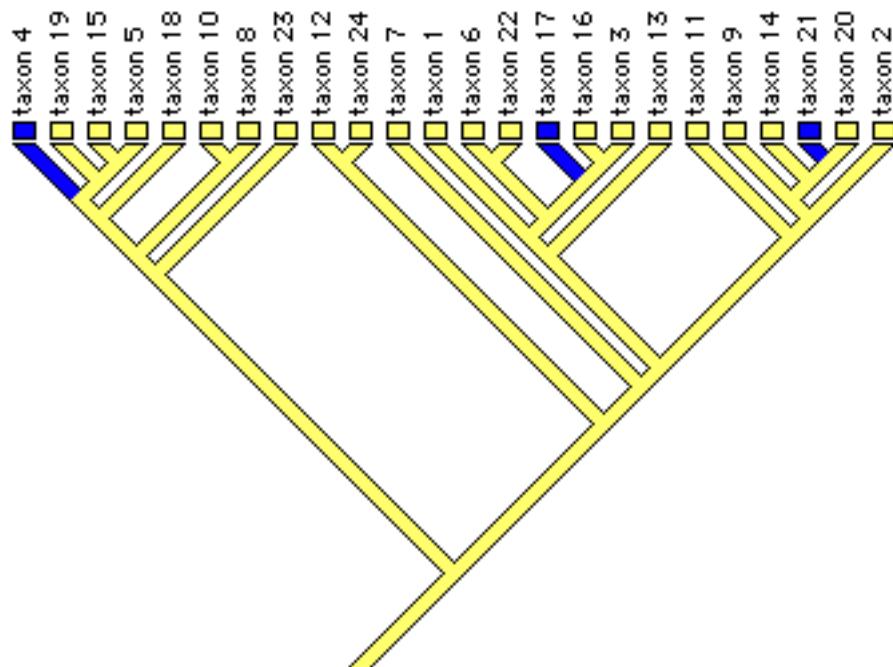
You can ask what is the probability of having more than, as many as, or fewer than an indicated number of gains and losses in the area distinguished in the traced character, given the number of gains and losses over the whole clade specified at the top, and given that 0 or 1 is assumed ancestral in the character of interest. For instance, if you want to ask what is the probability of having 4 or more gains (and any number of losses) in the distinguished area, given 5 gains 2 losses overall, then you should indicate 5 gains 2 losses at the top, select "either" for ancestral state; indicate "more" and "as many" and "4" under gains; and "more", "as many", and "fewer" under losses.

When the exact calculations are used, the **Correlation Test Results** dialog box allows you to change the total number of gains and losses at the top, as long as the higher of the number of gains or losses does not exceed the higher of the number of gains or losses you entered for the total in the lower left of the **Correlation Test Parameters** dialog box. With simulations, the **Correlation Test Results** dialog box does not allow you to change the overall gains and losses.

If you click on Output to file, a text file will be written showing the probability of various numbers of gains and losses occurring in the distinguished area, given the number of gains and losses overall, and the ancestral state that you have indicated at the top of the dialog box.

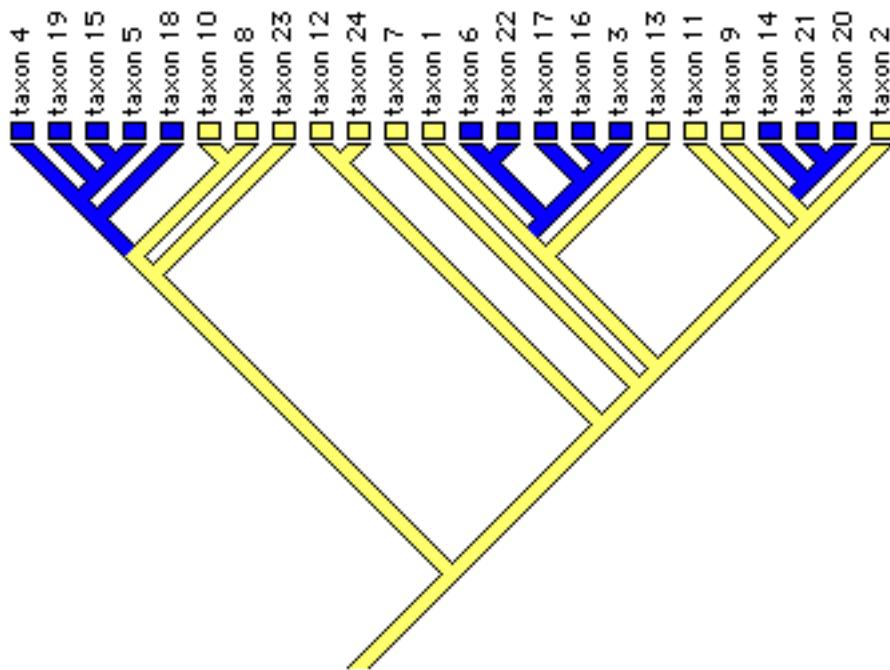


EXAMPLE: Consider the following phylogeny, in the file "Concentrated Changes Example", with character 1 traced upon it:



You note that "black" has arisen three separate times. Upon examining character 2, you notice that the three origins

of black in character 1, above, all occur within groups that have black in character 2:



This suggests that there might be some sort of correlated evolution of characters 2 and 1; perhaps something about character 2 being white prevents the evolution of black in character 1; when this constraint is freed, black can evolve in character 1.

But are the three changes in character 1 overly concentrated in the black regions in character 2? MacClade's concentrated-changes test allows you to test the following hypothesis: The changes in character 1 are randomly distributed about the clade selected. To do this, we ask the question: What is the probability, under this null hypothesis, that three gains of black in character 1 would occur on black lineages in character 2?

 With character 2 traced, click the character-correlation tool () on the root of the tree. In the dialog box that comes up, type 3 for gains and 0 for losses, and click on OK (exact calculations will be done). In the dialog box that comes up after calculations are done, specify as many or more than three gains, and 0 losses. The probability listed is 0.120671, the probability of having all three gains fall in the black area were they distributed by chance.

Appropriateness of the concentrated-changes test

W. Maddison (1990) discusses some of the drawbacks of the concentrated-changes test. One is the assumption that all branches are alike: the null hypothesis supposes change is equally likely on each branch. W. Maddison (1990) discusses situations in which this assumption could cause problems. If this assumption is not reasonable, then you might better construct simulations that consider branch lengths, or consider an alternative test such as that of Pagel (1994). Another problem discussed by W. Maddison is the difficulty of separating an effect of the independent variable from a clade-specific effect when most of the changes are concentrated in one clade. This problem is highlighted by Read and Nee (1995), who suggest pairwise comparisons as an alternative. Pairwise comparisons can be a useful alternative but even they can have analogous problems (Ridley and Grafen, 1996; W. Maddison, 2000).

The reader is directed to Lorch and Eadie (1999) for an investigation of the power of the test, to Sillén-Tull-

berg (1993) for an alternative but related test, and to Pagel (1994) for a likelihood-based test.

Confirming the validity of the correlation test algorithms

In programming MacClade, the proper functioning of the correlation-test algorithms was tested as follows.

- The source code was thoroughly checked and rechecked.
- The exact and simulations calculations provide a natural cross-check for each other. For numerous test cases, simulations calculations were shown to provide accurate estimates of the exact results.
- Both simulations and exact calculations were shown to give correct results in a few small test cases that could be calculated by hand.
- The simulations keep track of how many changes were assigned to each branch. Each branch should get approximately the same number of changes assigned. If more than 100 changes are assigned to at least one branch, and another branch has less than half as many, then a warning is given. This is a weak test but is effective if the algorithm is misbehaving badly. In addition, if you save the results to a file, then included in the file is a branch-by-branch list of how many changes were assigned. In test cases, the changes were assigned evenly enough.
- The probability shown on the **Correlation Test Results** dialog box when it first appears to the user should be 1. In tests it was consistently close enough to 1.



GENERATING RANDOM DATA AND RANDOM TREES

23

MacClade has several tools for generating data or trees randomly. These could be used, for example, to determine if the observed values of some statistic differ from those obtained under a particular random model. These tools all use a random number generator, which will be described first.

Random number generator and its seed

To generate random data or trees, MacClade uses a random number generator. This generator begins with a number called a "seed", and from it calculates a pseudo-random number. If it is to generate more than one random number, MacClade uses the previous random number as the seed for the next step. Given the same starting seed, the generator will generate the same sequence of numbers.

The random number generator is the same as in PAUP 3 (Swofford, 1991). It is a linear congruential random number generator, where the next number x_i is calculated from the previous in the sequence x_{i-1} by the formula $x_i = 397204094 * x_{i-1} \bmod (2^{31} - 1)$. (For discussion of the random number generator, see Payne, Rabung, and Bogyo, 1969; Fishman and Moore, 1982; Swofford, 1991.)

If left to itself, the generator will choose its own starting seed, based on the current time and date. To see or change the seed used by MacClade (e.g., to make a sequence of random data or trees reproducible), press

the seed button () , present in any of the **Random Trees** or **Random Data** dialog boxes. The following dialog box will appear:



You can set a new seed with the dialog box. Setting the seed to the seed used in a previous randomization will allow you to duplicate the previous randomization exactly (with the exception noted below). We suggest that you touch the seed button to see and record the seed before you begin a randomization, in case you want to duplicate the randomization exactly.

There is one circumstance in which it will be difficult to duplicate a randomization. This occurs when you are making random dichotomous resolutions of a polytomous tree on the screen. Even if you have the same polytomous tree on the screen, and set the seed to be the same, regenerating a set of random dichot-

omous resolutions may result in a different set each time. The reason for this is that MacClade stores polytomous trees internally as being dichotomous with some nodes marked as being "ghost" nodes that don't actually exist. This hidden "ghost" dichotomous resolution affects the random dichotomous resolutions obtained, and at different times you can have different ghost resolutions residing inside a polytomy. The best way to eliminate variations in this ghost resolution is to choose **Store Tree**, then choose **Trees** from the **Trees** menu to get the polytomous tree from storage just before use, because on processing from storage, the hidden resolution in a polytomy is always a ladder-like arrangement.

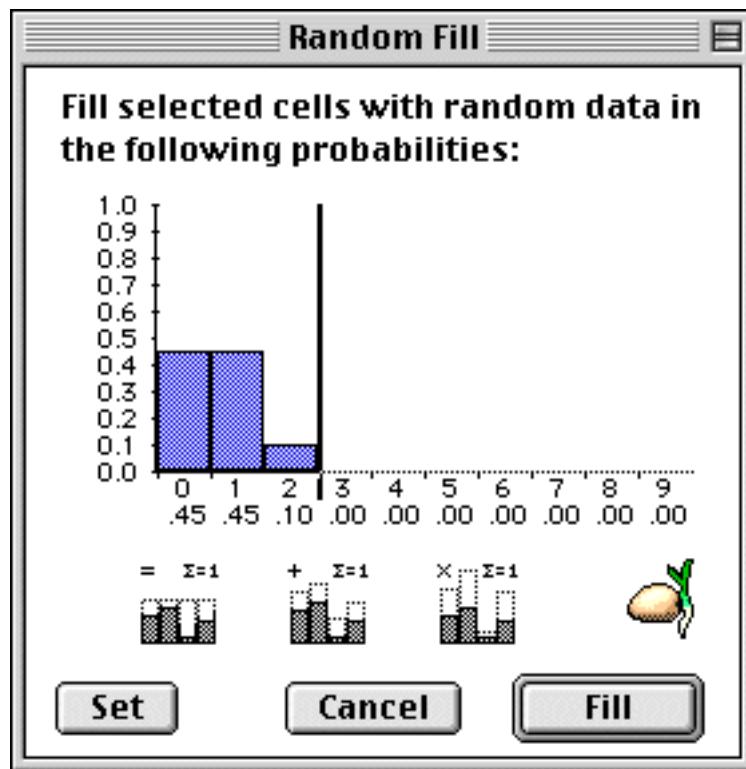
Random data

MacClade can generate new random data, or it can randomly shuffle existing data. Data can be generated randomly in two ways: either states are assigned randomly to the taxa without reference to a tree, or states are evolved using a stochastic model of change on the current tree.

Randomly assigning states to data cells

You can randomly assign states to cells using one of two methods: (1) the more-flexible fill states tool (see [page 233](#)), or (2) using the **Random Fill** dialog box. The **Random Fill** dialog box will be described here in this section.

To use the **Random Fill** dialog box, select a portion of the matrix in the editor and choose **Random Fill** from the **Utilities** menu; you will be presented with a dialog box in which you can set frequencies of states:



On the horizontal axis of the bar chart in the dialog box are the states, indicated by their current symbols (in the example shown above, 0 – 9). Each cell will be filled with a state with the probabilities shown on the bar chart. In the example illustrated above, each cell would be filled with either 0, 1, or 2, with probabilities 0.45, 0.45, and 0.10. To alter the frequency with which a state will be generated, move the top of the bar for that state up or down. Move the black vertical bar to indicate the maximum state. This is necessary to

include states of higher value; for example, in the above example, if you wanted to set the probability of state 6 to be 0.2, you would first have to move the black vertical bar over to the right of state 6. It is also important for the norming functions, as described in the next paragraphs.

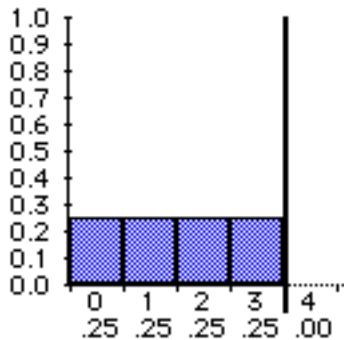
Remember, the frequencies must sum to 1! The three **norming buttons** can be used to make frequencies sum to 1. They have the following actions:

| Norming button | Action |
|---|---|
| $= \Sigma=1$  | Equal-frequencies norming button. Makes frequencies equal (from state 0 through to the maximum state indicated by the vertical bar) |
| $+ \Sigma=1$  | Scales frequencies to sum to 1 by adding or subtracting a constant amount to each |
| $\times \Sigma=1$  | Scales frequencies to sum to 1 by multiplying or dividing, thus maintaining their proportions |

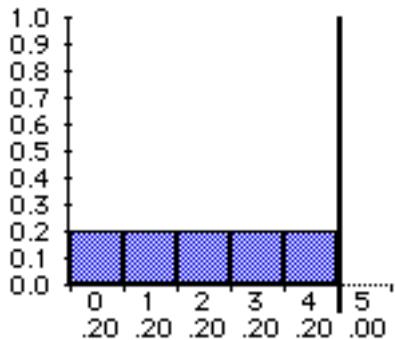
The norming buttons act on all states between 0 and the vertical black bar. For example, pressing on the

$= \Sigma=1$

equal-frequencies norming button () with the vertical bar set to be just above state 3 will set states 0 to 3 at equal frequency, as shown below on the left. If the vertical bar is just above state 4, then states 0 through 4 will be set to have equal frequency (below, right).



Effect of the equal-frequencies norming button with vertical bar above state 3



Effect of the equal-frequencies norming button with vertical bar above state 4

Once you have chosen the desired frequencies, press **Fill** to fill the selected cells with random data.

If you wish to set the desired frequencies for later use, but not fill any cells, then press **Set** once you have set the frequencies.

The fill states tool ([page 233](#)) provides more options for random data than does the **Random Fill** dialog box.

Shuffling existing data

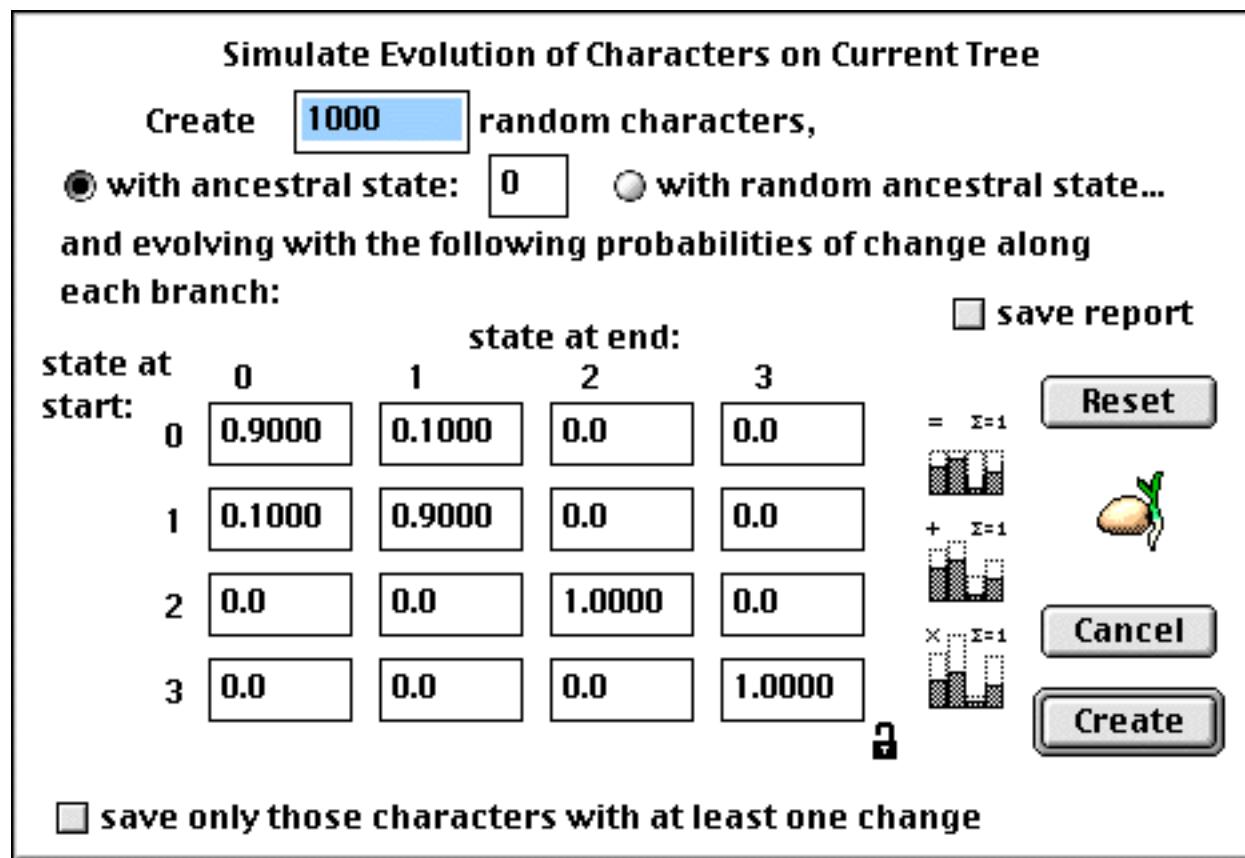
To randomly shuffle existing data, select a portion of the matrix and choose **Shuffle** from the **Utilities** menu in the data editor. MacClade will scramble the selected portion. For each character, the states assigned to the taxa in the selected portion are scrambled so that they are now assigned randomly to (possibly different) taxa within the selected portion.

When you choose **Shuffle**, a dialog box is presented, letting you choose the random number seed, and asking if you wish to remove all footnotes and pictures from the shuffled block of data. If you choose not to remove footnotes and pictures, they will not be shuffled, and will remain stuck to the same cell in the editor they were attached to before shuffling.

This data shuffling can be used, for example, to calculate Archie's (1989a,b) homoplasy excess ratio.

Evolving characters up a tree

To generate data by a stochastic model of change on the current tree, you need to be in the tree window. Select **Evolve Characters** in the **Edit** menu. The following dialog box allows you to enter the parameters of the stochastic model.



MacClade will randomly evolve the requested number of characters, according to the model, and add these on to the end of the data matrix. The probabilities shown indicate, for instance, that if a branch begins with state 0, there is a 0.1 = 10 % probability the lineage will be at state 1 at the end of the branch.

Press Create to create the characters.

If you choose "save only those characters with at least one change", then characters that did not change during their simulated evolution will be discarded.

While MacClade is evolving these characters, it keeps track of the actual changes between states that take place in the simulated evolution. If you have checked "save report", then when MacClade has finished creating all of the characters, a dialog box will come up, asking if you wish to save this information and other aspects of the simulation to a text file.

You can modify the model of evolution used, as described in the following sections.

Norming the frequencies in each row

The buttons on the right-hand side are the norming buttons; that is, they will cause the probabilities to be adjusted so that the sum in each row is 1.00. The norming buttons by default adjust all values in a row, including the value on the diagonal. If you want the values on the diagonals to be unaffected by the norming buttons, you can lock the diagonals by clicking on the lock at the lower-right corner of the matrix. If you click on the lock again, the diagonals will be unlocked.

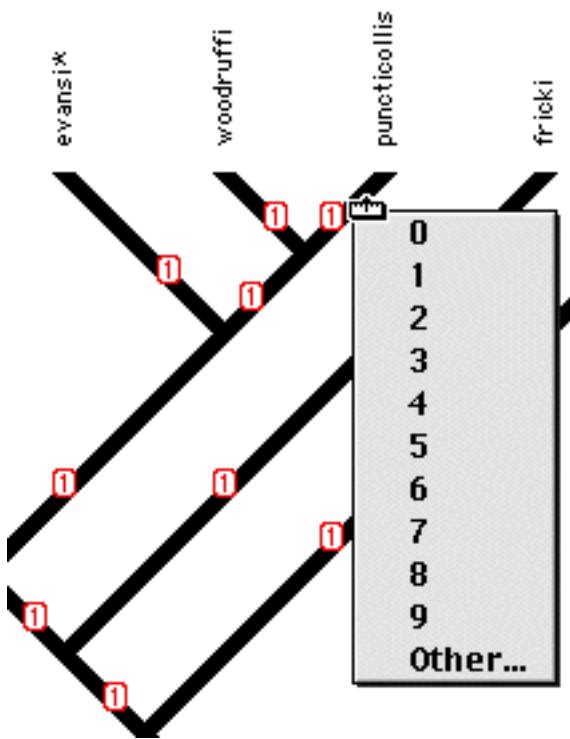
The norming buttons work in the following ways:

| Norming button | Action |
|---|---|
|  | Makes frequencies within in each row of the probability matrix equal. If the diagonal elements of the matrix are not locked, then this button will set all probabilities to 0.25. If the diagonal is locked, then this button will set the three off-diagonal probabilities in each row to equal $(1 - \text{prob diagonal element})/3$. For example, if the probability of a 0 to 0 change is set to 0.70, and the diagonals are locked, then this norming button will set the probabilities of 0 to 1, 0 to 2, and 0 to 3 all equal to 0.10. |
|  | Scales frequencies in each row to sum to 1 by multiplying or dividing, thus maintaining their proportions. |
|  | Scales frequencies to sum to 1 by multiplying or dividing, thus maintaining their proportions |

Setting branch lengths: Evolve segments

By default, there is an equal probability of change along each branch in the tree. You can make some branches longer than others, thus increasing the probability of change, by increasing the number of **segments** on the branches. On branches with more than one segment, MacClade will perform one round of evolution (with the probabilities chosen in the dialog box above) along each segment of the branch. The dividing of a branch into segments is a special way to indicate branch length that is used currently only by the Evolve Characters feature. It is not connected to the branch lengths arising from Trace All Changes and is not used in the correlation test nor in reconstructing the evolution of continuous characters.

You can set branches to have more than one segment, using the Set Evolve Segments tool (). With this tool, click on a branch, and choose a value from the pop-up menu, as shown below:



With this menu you can set the number of segments on the branch. (You can see the number of segments on each branch by choosing **Show Evolve Segments** from the **Display** menu.)

This tool has three modes, available in the tool's pop-up menu:



Normal Mode is as described above. In **Assign 1** mode, the tool will set the number of segments on all branches in the clade above the branch clicked to one. In **Assign Random** mode, all branches in the clade will be randomly assigned between 1 and 9 segments, with equal probability.

The number of segments that can be assigned to a branch for the **Evolve Characters** feature cannot be more than 1000.

WARNING: *The numbers of segments are saved only for the tree in the tree window (other trees stored in the file are not stored with their numbers of segments). However, in special cases in which a tree change is Undone, all evolve segments may revert to 1. Therefore if you are using Evolve Characters with segments, you should check to see that the segments are as you want them just before using them.*

If some branches have more or less than one segment, then a check box will appear in the **Evolve Characters** dialog box, entitled "consider branch segments". If this is checked, then the probabilities given in the matrix will be per branch segment, not per branch, so that on a branch with three segments MacClade will perform three rounds of evolution on the branch. Thus, a 3-segmented branch might start out with state 0,

change to state 1 on the first segment, remain at state 1 through the second segment, and then change to state 2 on the third segment.

Ancestral state

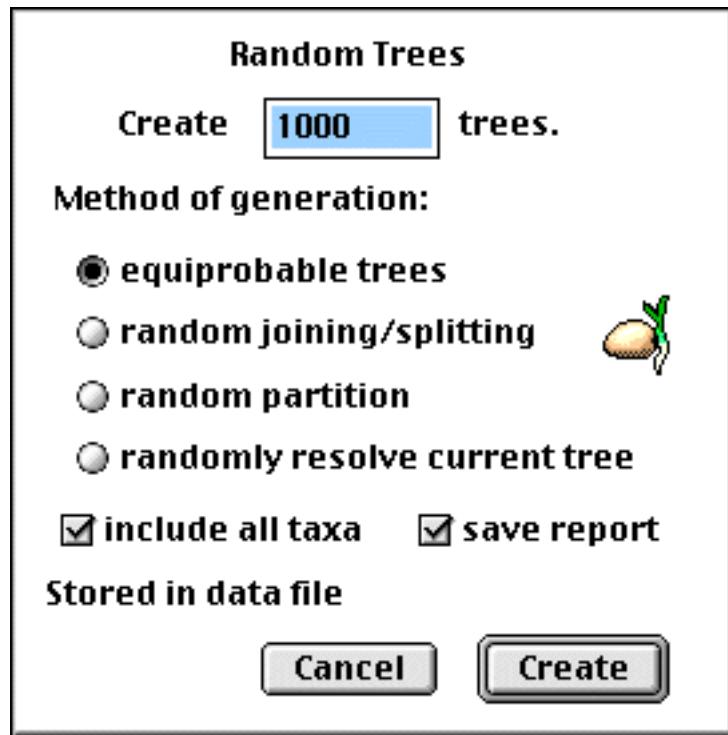
The ancestral state for each created character will be 0 or another fixed state chosen by you, or you can ask MacClade to generate ancestral states randomly according the frequencies you set when you click on "with random ancestral state". The dialog box in which you set the ancestral frequencies behaves exactly like the one described in the preceding section.

Confirming the validity of the random data algorithms

In programming MacClade, the proper functioning of the random data algorithms was tested by comparing the outcome of the randomization with the parameters of the model used; several different parameter values were tested. The frequency of states produced by **Shuffle** and **Random Fill** match those expected, as calculated by the States chart, and the frequency of changes produced by Evolve Characters in the tree window as written in the Simulation Report file match those of the model.

Random trees

MacClade can generate random dichotomous trees in various contexts, for instance in charts. MacClade can also generate and save random trees into the current data file or tree file, if you choose **Random Trees** from the **Trees** menu.



The Random Trees dialog box

There are four options for generating trees. The first three of these relate to different probability distributions of completely random trees (Maddison and Slatkin, 1991):

Equiprobable trees: The probability of picking any given tree out of all possible dichotomous rooted trees is equal. Random trees so generated sample evenly all possible dichotomous trees. This might be useful, for instance, to examine the shape of the bar chart of treelengths over all possible trees (e.g., Fitch, 1979; Helsenbeck, 1991), without actually enumerating all possible trees.

Random joining: Trees are formed by successively joining taxa randomly. Thus, two terminal taxa are chosen at random and made into a new clade, and this clade is treated as if it were a terminal taxon for purposes of further joining. The taxa are randomly joined until all have been clustered. Such random joining trees are useful to model the coalescent process in a panmictic population (see example, [page 411](#)), or to model a random speciation process in which each species has an equal chance of speciating (Harding, 1971; Simberloff et al., 1981; Maddison and Slatkin, 1991).

Random partition: Trees are formed by successive random partitions of the taxa. A random partition of all taxa into two subsets is first made, with each terminal taxon having a 0.5 chance of falling into the left-hand subset. Then within each subset, again a random partition is made, and so on until a fully resolved tree is created. Such random partitioning trees might be useful for examining trees generated by a divisive clustering method whose decisions are taken randomly. Although no biological process has been proposed that yields tree distributions of this sort, random partition trees provide a heuristic extreme that emphasizes symmetrical trees.

Random dichotomous resolution of polytomous tree: There is a fourth option that does not create a fully random tree. It appears only when the tree in the tree window has polytomies, and it allows the user to create random dichotomous resolutions of this polytomous tree. The tree on screen is kept as is, except that a dichotomous resolution of each polytomy is chosen randomly (equiprobably for each possible resolution) to yield a dichotomous tree. This option is useful for sampling from among all possible dichotomous resolutions consistent with a given polytomous tree. It is discussed further in the section on "[Looking at alternative resolutions" on page 108](#), although such dichotomous resolutions are also accessible in the other contexts in which random trees are generated.

Comparatively, samples of random partition trees are more biased toward symmetrical trees and equiprobable trees toward asymmetrical trees. Random joining trees are intermediate in their bias.

If taxa are excluded from the current tree on the screen, and the "include all taxa" box is not checked, then trees will be formed of only the taxa in the current tree (all taxa in the tree, not just those visible if a clade happens to be expanded). If the "include all taxa" box is checked, then all taxa in the data matrix will be included in each random tree.

Just above the Create button, MacClade lists where the created trees will be stored — in the current tree repository, either the data file or a external tree file — if they are to be stored.

Confirming the validity of the random tree algorithms

In programming MacClade, the proper functioning of the random tree algorithms was tested as follows.

- The source code was thoroughly checked and rechecked.
- MacClade will keep track of the basal split in the tree and compare observed basal clade size asymmetries with those theoretically expected for the type of random tree being generated. Theoretical expectations are given by the equations for R given by Maddison and Slatkin (1991), except that the equation for equiprobable trees is not correctly listed in their paper: each expression should be divided by 2. (The results of their paper were calculated using the correct expressions.) In test cases, MacClade gave observed asymmetries very close to those expected. If you want to

use this check, select "save report" in the dialog box where you request random trees to be generated.

- The random tree algorithms yielded the appropriate probability distributions expected in theory for the number of steps in characters (Maddison and Slatkin, 1991).

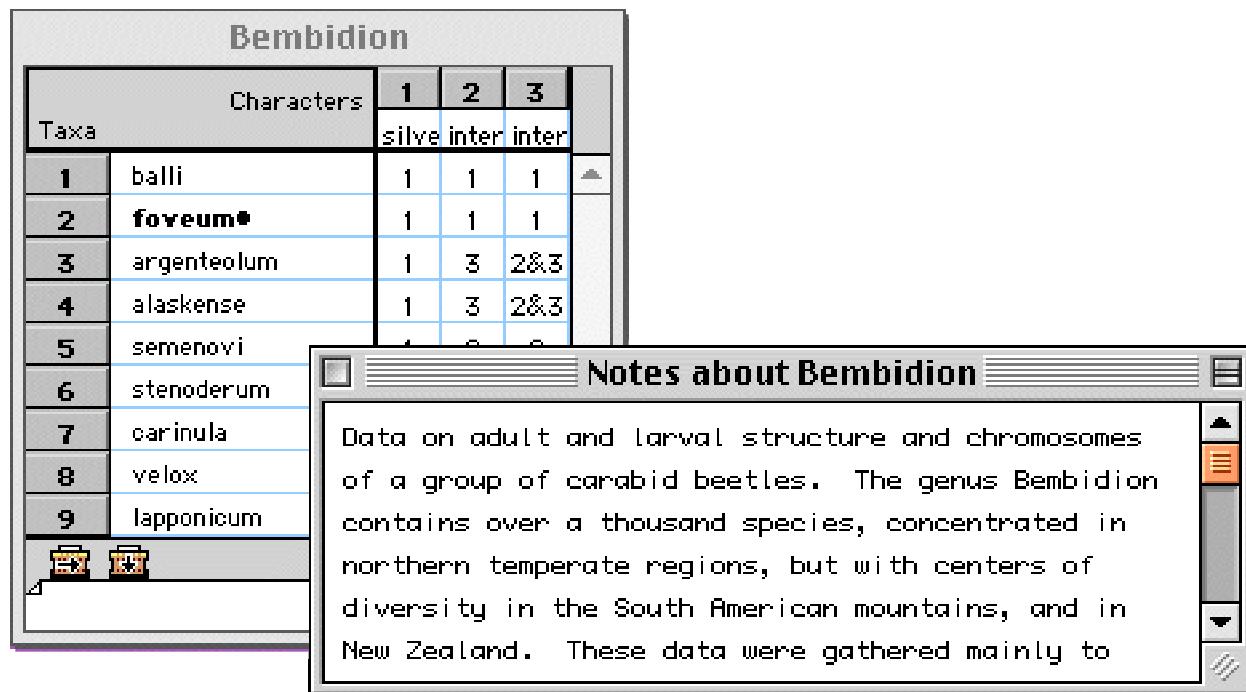


NOTE KEEPING

24

Adding notes about the data file

You can enter a general comment about the data file as a whole in the window that appears when you choose **File Notes** from the **Windows** menu:



You can, for example, include information about the source of the data, literature citations, or instructions to students. You may also wish to include notes about characters and taxa, work that needs to be done with the data, and so on. All of the example data files included with MacClade have notes added to them in the file notes window. (Notes added to this window will appear in PAUP*'s main window as the file is being executed.)

Adding notes about the trees

You can save notes and comments about the trees; use the **Notes About Trees** window, available from the **Windows** menu. These notes are stored with the trees in the current tree repository (that is, either in the data file or tree file). If you do a tree search in PAUP* and save the trees to a file, PAUP* will store information in the tree file about the source of the trees; this information will appear in MacClade's **Notes About Trees** window if you open that tree file.

Annotating cells in the data matrix

You can store text notes or pictures for each cell in the editor.

Footnotes

MacClade will allow you to store footnotes to the cells in the matrix. For example, in the matrix below, character 2 for taxon 5 has a footnote attached to that cell, and because this cell is currently selected, its footnote is displayed in the thin strip along the bottom of the window.

| | | Characters | 1 | 2 | 3 | 4 | 5 | 6 | 7 | |
|---|----------------|------------|-------|-------|-------|-------|-------|--------|-------|--|
| | | Taxa | silve | inter | inter | inter | outer | silver | elytr | |
| 1 | balli | | 1 | 1 | 1 | 0 | - | 0 | 0 | |
| 2 | foveum* | | 1 | 1 | 1 | 0 | - | 0 | 0 | |
| 3 | argenteolum | | 1 | 3 | 2&3 | 0&1 | - | 0 | 0 | |
| 4 | alaskense | | 1 | 3 | 2&3 | 0 | - | 0&1 | 0 | |
| 5 | semenovi | | 1 | 2 | 2 | 0 | - | 0 | 0 | |
| 6 | stenoderum | | 1 | 3 | 3 | 0&2 | 0 | 0 | 0 | |
| 7 | carinula | | 1 | 3 | &1&2 | 0 | - | 0 | 0 | |

T 10 C 8

1 specimen examined; check this on more, if possible!

A footnote displayed for taxon 5, character 2

When a cell with a footnote is not selected, the presence of a footnote is indicated by an asterisk:

| | | Characters | 1 | 2 | 3 | 4 | 5 | 6 | 7 | |
|---|----------------|------------|-------|-------|-------|-------|-------|--------|-------|--|
| | | Taxa | silve | inter | inter | inter | outer | silver | elytr | |
| 1 | balli | | 1 | 1 | 1 | 0 | - | 0 | 0 | |
| 2 | foveum* | | 1 | 1 | 1 | 0 | - | 0 | 0 | |
| 3 | argenteolum | | 1 | 3 | 2&3 | 0&1 | - | 0 | 0 | |
| 4 | alaskense | | 1 | 3 | 2&3 | 0 | - | 0&1 | 0 | |
| 5 | semenovi | | 1 | 2* | 2 | 0 | - | 0 | 0 | |
| 6 | stenoderum | | 1 | 3 | 3 | 0&2 | 0 | 0 | 0 | |
| 7 | carinula | | 1 | 3 | &1&2 | 0 | - | 0 | 0 | |

Presence of a footnote indicated by an asterisk in the cell for taxon 5, character 2

To activate the footnote system, make sure **Footnotes** is checked in the **Display** menu. Select the cell you

wish to footnote. Then click on the little message area at the very bottom of the data editor, and enter the footnote. Footnotes can be attached to each cell in the data matrix, as well as to the cells displaying taxon names and character names. If a cell has a footnote, the entry in that cell will be boldfaced and contain an asterisk. (Note that crocodiles have character 2 footnoted in the figure shown.)

By default, the footnote area has room for only one line of text, but this can be increased by dragging the top of the footnote box up, so that it has two or more lines, as shown below:

| | | Characters | | | |
|------|----------------|------------|---|-------------|--|
| | | 1 | 2 | | |
| Taxa | | | | silve inter | |
| 1 | balli | 1 | 1 | | |
| 2 | foveum* | 1 | 1 | | |
| 3 | argenteolum | 1 | 3 | | |
| 4 | alaskense | 1 | 3 | | |
| 5 | semenovi | 1 | 2 | | |

T 7 C 3

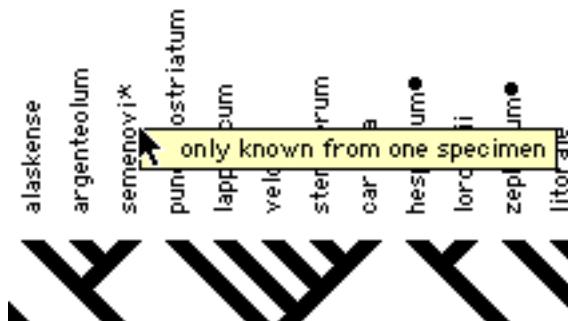
1 specimen examined; check this on
more, if possible.

The footnote bar expanded to two lines

NOTE: To clear the footnotes from a block of selected cells, hold down the Option key and then choose **Remove All Selected Footnotes** from the **Edit** menu.

NOTE: Option-↓ will pop the insertion point from a matrix cell down to the footnote, and Option-↑ will pop it back up to the matrix.

To show a footnote attached to a taxon name in the tree window, click on the taxon name, as shown below:



Taxa with footnotes are indicated by an asterisk ("*") in the taxon name. To show a footnote of a character state in a taxon, click the arrow tool on the box just beneath the taxon's name when the character is traced, or click on the appropriate box in the Data Boxes display; the box will be outlined in gray if a footnote is stored.

Pictures linked to taxa, character names, or data

Illustrations can be linked to the cells of the matrix and displayed on command. Thus you can have a fig-

ure depicting an individual of the taxon attached to the taxon name (as illustrated below), or a figure depicting the taxon's state in a particular character attached to the corresponding cell in the matrix. You might wish to store a map of the geographic distribution of a taxon, or a picture summarizing all the states of a character. An example is shown below.

| | | Bembidion | | | | | | | | | |
|------|--------------------|------------|-------|-------|-------|-------|-------|-------|-------|-------|-----|
| | | Characters | | | | | | | | | |
| | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| Taxa | | silve | inter | inter | inter | outer | silve | elytr | subor | midse | 3 - |
| 6 | stenoderum | 1 | 3 | 3 | 0&2 | 0 | 0 | 0 | 1&2 | 0 | 0 |
| 7 | carinula | 1 | 3 | &1& | 0 | - | 0 | 0 | 2 | 1 | 0 |
| 8 | velox | 1 | 3 | &1& | 1 | - | 0 | 0 | 2 | 1 | 0 |
| 9 | lapponicum | 1 | 3 | .1&2 | 0&2 | 0 | 0 | 0 | 2 | 1 | 0 |
| 10 | punctatostriatum | 1 | 3 | &1& | 1 | - | 0 | 0 | 0 | 1 | 0 |
| 11 | hesperium* | 1 | 2&3 | 3 | 0&2 | 0 | 0&1 | &1& | 1 | 0 | 0 |
| 12 | lorquinii | 1 | 3 | 3 | 1 | - | 0&1 | 0 | 0&1 | 0 | 0 |
| 13 | zephyrum* ♀ | - | - | - | &3 | 0 | 0&1 | 0 | 2 | 1 | 0 |
| 14 | I. levettei | 3 | 0 | 1 | 0 | 2 | 1 | 0 | | | |
| 15 | I. carriatum | 3 | 0 | 1 | 0 | 2 | 1 | 0 | | | |
| 16 | inaequale | 3 | 0&1 | 0&1 | 0 | &1& | 0&1 | 0 | | | |
| 17 | litorale | 3 | 0&1 | 0&1 | 0 | 2 | 0&1 | 0 | | | |
| 18 | conicolle | 3 | 0&1 | 0&1 | 0 | 2 | 0 | 0 | | | |
| 19 | chloropus | 0 | - | - | 0 | 0 | 0 | 0 | 1 | | |
| 20 | fusiforme | 0 | - | - | 0 | 0 | 0 | 0 | 1 | | |
| 21 | aeneipes | 0 | - | - | 0 | 0 | 0 | 0 | 1 | | |
| 22 | sp. nr. aenulum | 0 | - | - | 2 | 0 | 0 | 0 | 1 | | |
| 23 | aenulum | 0 | - | - | 0 | 0 | 0 | 0 | 1 | | |
| 24 | durangoense | 0 | - | - | 0 | 0 | 0 | 0 | 1 | | |
| 25 | arizonae | 0 | 0 | 0 | 0 | - | - | 2 | 0 | 0 | 1 |
| 26 | confusum | 0 | 0 | 0 | 0 | - | - | 2 | 0 | 1 | 1 |

To link a picture to a cell in the matrix, you can either paste it from the Clipboard or import it from a PICT file. To paste the picture from the Clipboard, first put the picture into the Clipboard (perhaps in a graphics application), then select a cell in the matrix, and choose **Paste Picture** in the **Edit** menu. To import a picture from a file, click on a cell, then choose **Import Picture** under **Edit**. In the dialog that appears, choose the name of the file that contains the graphics image. The graphics file must be in PICT format.

MacClade indicates that a picture is attached to a cell by adding a black circle (•) to the contents of the cell.

For various technical reasons, MacClade requires a fair amount of disk space and computer memory to display, add, and remove pictures. If there is not enough disk space, or the disk is locked, then MacClade will allow you to display pictures, but not allow you to add or remove any.

To display a picture attached to a cell, touch on the cell in the data matrix with the Show Picture tool

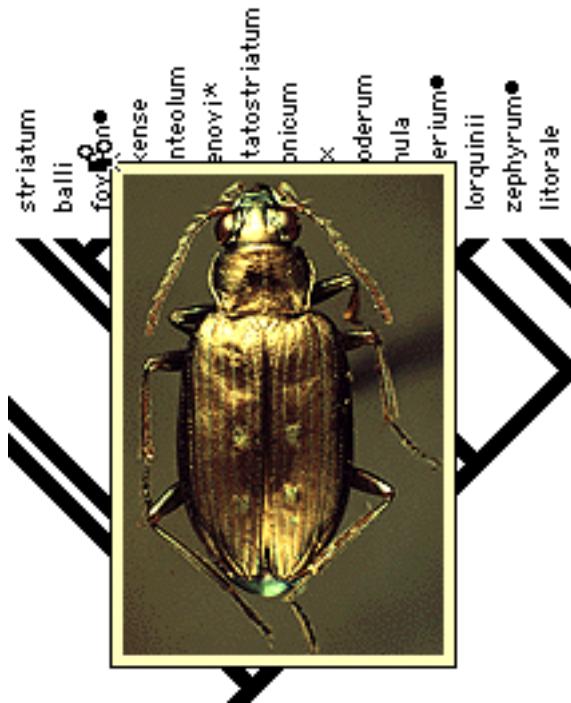
(below), or select the cell and choose **Show Picture** under **Display**. The moment you move to another cell, the picture window will disappear. Pictures can also be displayed from the tree window (see "[Show Picture tool](#)", below).

Pictures can be removed from a cell by selecting the cell and choosing **Remove Picture** from the **Utilities** menu. All pictures in a block of cells can be removed by selecting the block and choosing **Remove Pictures** from the **Utilities** menu.

WARNING: *If you have pictures stored in the data file, and edit and save the file with any program other than MacClade or PAUP*, you run the risk of losing all of the pictures. Before you edit files containing pictures using any application other than MacClade or PAUP*, we recommend that you try saving extra copies of the files first, then see if the pictures are still visible in MacClade. Beware that other alterations to the files may also result in loss of the pictures. For instance, some methods of moving text files along a network will lose the pictures unless you use binary transfer modes. When you open the file in PAUP* or a word-processing program, you will not see any evidence of the pictures. For the technically minded, the pictures are stored in the resource fork of the data file, as resources of type PICT. The reason that some word-processor programs discard the pictures is that they discard the resource forks of text files.*

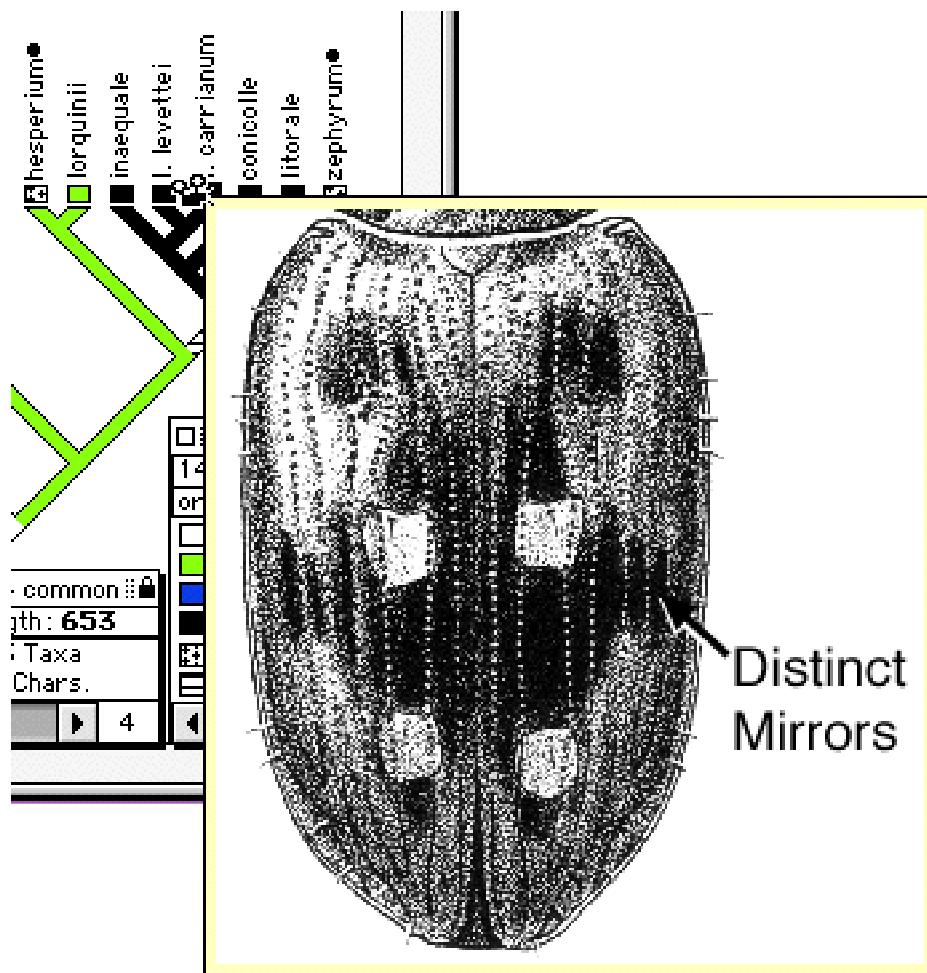
Show Picture tool

This tool, available in both the data editor and the tree window, allows you to display pictures attached to cells. In the data editor, it is used by clicking it on the cell containing the picture. By default, the picture will temporarily appear, and disappear once the mouse button is released. However, if you switch it to **Show Picture in Separate Window**, using the tool's pop-up menu, the picture will appear in a separate window. In the tree window the tool is used in a similar fashion. To show a picture attached to a taxon name, click the Show Picture tool on the taxon name, as shown below.

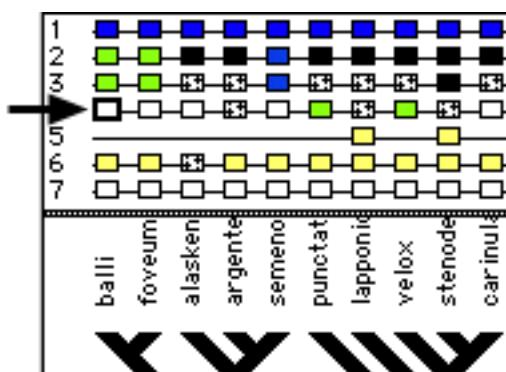


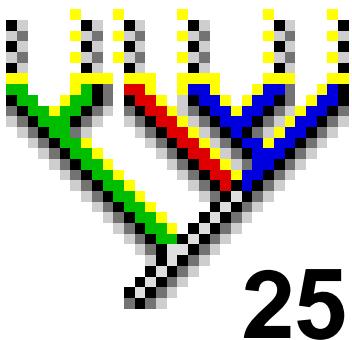
Taxa with pictures are indicated by a bullet ("•") in the taxon name in the tree window.

To show a picture of a character state in a taxon, click the Show Picture tool on the box just beneath the taxon's name when the character is traced, as shown below.



or click on the appropriate box in the Data Boxes display. The box will be outlined in black if a picture is stored (as indicated by the arrow below).





25

RECORDING YOUR WORK: PRINTING, GRAPHIC, AND TEXT FILES

Saving a record of your work

With some phylogenetic computer programs it is relatively easy to save a record of your work, because a formatted data file was processed by a program that generated an output tree. You need only print out the data file, the options in effect at the time of tree reconstruction, and a log file, and a complete record was saved. However, MacClade's interactive nature means that results large or small are constantly being generated against a changing background of assumptions. Although this encourages free exploration of alternatives, it makes saving a record of your work particularly difficult, especially as many of the results are presented in graphical, not textual, forms. MacClade's features for printing and saving graphics files provide the user with record-keeping ability, but it is largely up to the user to remember to keep records.

The first and most basic method to save records is to use **Save As** in the **File** menu to store archive copies of your data file (with its trees, assumptions, options, and so on) at critical points in the analysis. Remember, **Save As** will leave the last-saved version on disk with its old name and create a new file with the new name you specify. The new file will store the current options and data.

Second, various results, data, and assumptions can be printed. Trees and charts can be printed to store a record of results, and the data matrix to print a record of data. However, likely to be forgotten are the assumptions and options that can affect the results but that don't appear in the data matrix. The character list window, which contains a list of included characters, their weights and transformation types, can be printed, as can the other lists. A summary of current options and assumptions, transformation-type definitions, footnotes, pictures, genetic code, and any continuous data can be printed using the **Print Other** submenu in the **File** menu.

There is no facility in MacClade to print all aspects of the data file with a single command. For this reason, if you select a file in the Finder and give the **Print** command, MacClade will start, open that file, but not print (as MacClade doesn't know what aspect of that file you wish to print). If you want an archival print-out of the exact form of your data file as saved, you can open it in any word processor and print it (because MacClade files are stored as text files).

Third, images of trees and charts can be saved to graphical PICT files or PostScript files so that you can later edit and print them in graphics programs.

Fourth, general information about current assumptions, and tables such as the character list window and chart window can be saved as text files using the **Save Text File** submenu of the **File** menu. Text files can also store information generated by chart calculations, the character-correlation test, and by character tracings.

Printing the data matrix

There are three ways the data matrix can be printed: (1) by using **Print Matrix** from the **File** menu, (2) by choosing **Pretty Print Matrix** from the **File** menu, (3) by exporting the file to another format, which you then edit and print from another program. The first generates much cruder output, the second does not work well with some data, and the third requires use of another program.

The Print Matrix command

The data matrix can be printed when it is the frontmost window on the screen by choosing **Print Matrix** from the **File** menu. It will be printed more or less as it appears on the screen. It will therefore be rather crude. For example, choosing Print Matrix for the "Pretty Print Example" data matrix will yield three pages, each with about 30 characters shown for the three taxa, in one small strip at the top; many of the lines drawn on the data editor will also appear on this printout. The first page will look like this:



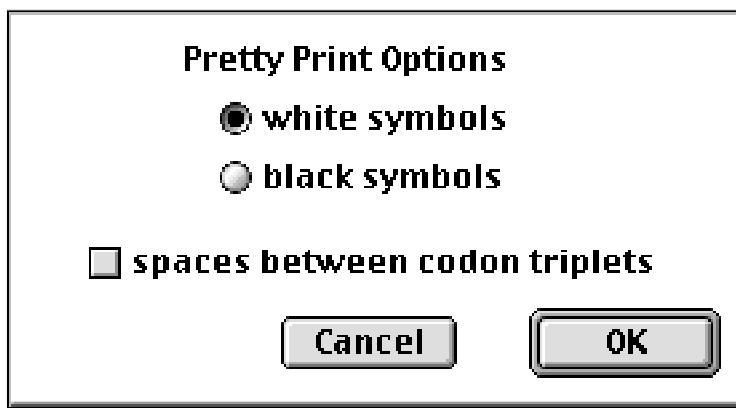
| Pretty Print Example | | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 |
|----------------------|-------------|---|---|---|---|---|---|---|---|---|---|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| 1 | Thylacinus | T | T | T | G | G | A | T | C | C | T | T | A | C | T | A | G | G | A | A | T | C | T | G | C | C | T | A | G | T | C | A | |
| 2 | Trichosurus | T | T | C | G | G | A | T | C | A | C | T | A | C | T | A | G | G | C | A | T | C | T | G | C | T | T | A | A | C | T | A | |
| 3 | Dasyurus | T | T | T | G | G | A | T | C | Y | C | T | A | T | T | A | G | G | A | G | T | A | T | G | C | C | T | A | A | T | T | A | |

The Pretty Print Matrix command

The **Pretty Print Matrix** command in the **File** menu is designed to print much more aesthetically pleasing matrices, but it does not work well for matrices with polymorphic cells or long state names shown. This command is available if the data editor is the frontmost window. The data will be printed in multiple blocks per page, without the lines shown in the data editor. For example, the Pretty Print Example matrix will print on one page, as follows:

| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----|-------------|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | Thylacinus | T | T | T | G | G | A | T | C | C | T | T | A | C | T | A | G | G | A | A | T | C | T | G | C | C | T | A | G | T | C | A | | | | |
| 2 | Trichosurus | T | T | C | G | G | A | T | C | A | C | T | A | C | T | A | G | G | C | A | T | C | T | G | C | T | T | A | A | T | C | C | | | | |
| 3 | Dasyurus | T | T | T | G | G | A | T | C | Y | C | T | A | T | T | A | G | G | A | G | T | A | T | G | C | C | T | A | A | T | T | A | | | | |
| 40 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 41 | Thylacinus | T | A | A | C | A | G | G | C | T | T | A | T | T | T | C | T | A | G | C | A | A | T | A | C | T | A | T | C | A | G | A | C | A | | |
| 50 | Trichosurus | T | T | A | C | A | G | G | C | T | T | A | T | T | T | C | T | A | G | C | A | A | T | G | C | T | T | G | A | T | A | C | A | | | |
| 60 | Dasyurus | T | C | A | C | A | G | G | C | C | T | A | T | T | T | C | T | A | G | C | A | A | T | A | C | C | T | T | C | A | G | A | C | A | | |
| 70 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 80 | Thylacinus | A | T | C | A | A | C | T | G | C | C | T | T | T | C | C | T | C | A | G | T | A | G | C | A | C | T | A | T | C | A | G | A | C | A | |
| 90 | Trichosurus | A | G | C | A | A | C | A | G | C | T | T | T | C | T | C | A | T | C | A | G | T | A | G | C | C | C | A | C | A | T | G | A | T | A | |
| 100 | Dasyurus | A | C | T | T | A | C | C | G | C | T | T | T | C | T | C | C | T | C | C | G | T | A | G | C | C | C | A | C | A | T | T | T | G | C | C |
| 110 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 115 | Thylacinus | A | T | C | A | A | C | T | G | C | C | T | T | T | C | C | T | C | A | G | T | A | G | C | A | C | T | A | T | C | T | G | T | C | | |
| | Trichosurus | A | G | C | A | A | C | A | G | C | T | T | T | C | T | C | A | T | C | A | G | T | A | G | C | C | C | A | C | A | T | G | A | T | A | |
| | Dasyurus | A | C | T | T | A | C | C | G | C | T | T | T | C | T | C | C | T | C | C | G | T | A | G | C | C | C | A | C | A | T | T | T | G | C | C |

When you choose **Print Pretty Matrix**, you will be presented with a dialog box in which you can choose whether the data are shown with white symbols or black symbols, and (if the data are protein-coding nucleotides) whether to put spaces between the codons:



You can also control the printout by changing the appearance in the editor, as described in the [Chapter 14](#). For example, if you ask to have every third position in codons lightened using the **Shade Char Sets** submenu, every third position will be pale not only in the editor but also in pretty printing:

| | | | | | | | | | |
|-------------|----|---|----|---|-----|---|-----|---|-----|
| | 1 | | 10 | | 20 | | 30 | | 40 |
| Thylacinus | T | T | G | G | A | T | C | C | T |
| Trichosurus | T | T | C | G | G | A | T | C | A |
| Dasyurus | T | T | T | G | G | A | T | C | T |
| | | | | | | | | | |
| | 41 | | 50 | | 60 | | 70 | | 80 |
| Thylacinus | T | A | A | C | A | G | G | C | T |
| Trichosurus | T | T | A | C | A | G | G | C | T |
| Dasyurus | T | C | A | C | A | G | G | C | T |
| | | | | | | | | | |
| | 81 | | 90 | | 100 | | 110 | | 115 |
| Thylacinus | A | T | C | A | A | C | T | G | T |
| Trichosurus | A | G | C | A | A | C | A | G | T |
| Dasyurus | A | C | T | T | A | C | C | G | C |

If you ask to have the cells in the editor colored by the amino acid into which that codon would be translated, those colors will also be used in pretty printing:

| | | | | | |
|-------------|---|----|-----|-----|-----|
| | 1 | 10 | 20 | 30 | 40 |
| Thylacinus | T T T G G A T C C C T T A C T A G G A A T C T G C C T A G T C A T T C A A A T C T | | | | |
| Trichosurus | T T C G G A T C A C T A C T A G G C A T C T G C T T A A C T A T A C A A A T C C | | | | |
| Dasyurus | T T T G G A T C Y C T A T T A G G A G T A T G C C T A A T T A T T C A A A T C C | | | | |
| | 41 | 50 | 60 | 70 | 80 |
| Thylacinus | T A A C A G G C T T A T T T C T A G C A A T A C A T T A C A C A T T C A G A C A C | | | | |
| Trichosurus | T T A C A G G C T T A T T C T T A G C A A T G C A C T A C A C A C C G C T G A T A C | | | | |
| Dasyurus | T C A C A G G C C T A T T C C T A G C A A T A C A C A C T A C A C T T C A G A C A C | | | | |
| | 81 | 90 | 100 | 110 | 115 |
| Thylacinus | A T C A A C T G C C T T C T C C T C A G T A G C A C A T A T C T G T T C | | | | |
| Trichosurus | A G C A A C A G G C T T C T C A T C A G T A G C C C A C A T C T G T T C | | | | |
| Dasyurus | A C T T A C C G C T T C T C C T C C G T A G C C C A C A T T T T G C C C | | | | |

If you ask the same printout to use black symbols and to show spaces between codon triplets, the result will be:

| | | | | | |
|-------------|---|-----|-----|----|----|
| | 1 | 10 | 20 | 30 | 33 |
| Thylacinus | T T T G G A T C C C T T A C T A G G A A T C T G C C T A G T C A T T C A A A T C T | | | | |
| Trichosurus | T T C G G A T C A C T A C T A G G C A T C T G C T T A A C T A T A C A A A T C C | | | | |
| Dasyurus | T T T G G A T C Y C T A T T A G G A G T A T G C C T A A T T A T T C A A A T C C | | | | |
| | 34 | 40 | 50 | 60 | 66 |
| Thylacinus | C A A A T C T T A A C A G G C T T A T T C T A G C A A T A C A T T C G C A A T A C A T | | | | |
| Trichosurus | C A A A T C C T T A C A G G C T T A T T C T A G C A A T G C A A T T G C A A T G C A C | | | | |
| Dasyurus | C A A A T C C T C A C A G G C C T A T T C T A G C A A T A C A C A T T T G C A A T A C A C | | | | |
| | 67 | 70 | 80 | 90 | 99 |
| Thylacinus | T A C A C A T C A G A C A C A T C A G C C T T C A T C T G C C T T C T C C T C A | | | | |
| Trichosurus | T A C A C C G C T A C C G A T A C A C A G C A T C T G C T T C T C T C A T C A T C A | | | | |
| Dasyurus | T A C A A C T T C A G A C A C A C T T C T G C T T C T C T C C T C A T C A T C A | | | | |
| | 100 | 110 | 115 | | |
| Thylacinus | G T A G C A C A T A T C T G T T C | | | | |
| Trichosurus | G T A G C C C A C A A T C T G T C | | | | |
| Dasyurus | G T A G C C C C A C A A T T T G C C | | | | |

Protein matrices can also be printed in this way, and their appearance can also be controlled similarly, for example by asking to use the three-letter amino acid designations in the cells:

| | 1 | | | | | | | | | | | 10 | | | | | 15 |
|-------------|-----|-----|-----|-----|-----|-----|------|-----|-----|------|------|-----|------|-----|-----|--|----|
| Thylacinus | Phe | Gly | Ser | Leu | Leu | Gly | Ileu | Cys | Leu | Val | Ileu | Gln | Ileu | Leu | Thr | | |
| Trichosurus | Phe | Gly | Ser | Leu | Leu | Gly | Ileu | Cys | Leu | Thr | Met | Gln | Ileu | Leu | Thr | | |
| Dasyurus | Phe | Gly | Ser | Leu | Leu | Gly | Va | Cys | Leu | Ileu | Ileu | Gln | Ileu | Leu | Thr | | |
| | 16 | | | | | | | | | | 20 | | | | | | 30 |
| Thylacinus | Gly | Leu | Phe | Leu | Ala | Met | His | Tyr | Thr | Ser | Asp | Thr | Ser | Thr | Ala | | |
| Trichosurus | Gly | Leu | Phe | Leu | Ala | Met | His | Tyr | Thr | Ala | Asp | Thr | Ala | Thr | Ala | | |
| Dasyurus | Gly | Leu | Phe | Leu | Ala | Met | His | Tyr | Thr | Ser | Asp | Thr | Leu | Thr | Ala | | |
| | 31 | | | | | | | | | | 38 | | | | | | |
| Thylacinus | Phe | Ser | Ser | Val | Ala | His | Ileu | Cys | | | | | | | | | |
| Trichosurus | Phe | Ser | Ser | Val | Ala | His | Ileu | Cys | | | | | | | | | |
| Dasyurus | Phe | Ser | Ser | Val | Ala | His | Ileu | Cys | | | | | | | | | |

A morphological matrix, with states colored if they match the state in the first taxon, would pretty print as follows:

| | 1 | | | | | | | | | | | 10 | | | | | 20 | | | |
|----------------|---|---|---|---|---|---|---|---|---|---|---|----|---|---|---|---|----|---|---|---|
| seriatoporus-g | 1 | 0 | 1 | 0 | 3 | 3 | 1 | 0 | 0 | 3 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 |
| ellipticus-g | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 |
| amaroides-g | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 5 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 |
| ruficollis-g | 0 | 1 | 0 | 2 | 3 | 1 | 1 | 0 | 0 | 6 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 |
| iripennis | 0 | 1 | 0 | 1 | 1 | 3 | 2 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 0 |
| mexicanus | 0 | 1 | 0 | 0 | 2 | 2 | 2 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 2 | 0 |
| interior | 0 | 1 | 0 | 0 | 2 | 2 | 2 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 2 | 0 |
| evansi | 0 | 1 | 0 | 0 | 0 | 2 | 2 | 2 | 0 | 1 | 3 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 1 |
| panamensis | 0 | 1 | 0 | 1 | 2 | 2 | 2 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 0 |
| fricki | 0 | 1 | 0 | 1 | 2 | 2 | 2 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 2 | 0 | 1 | 0 |
| woodruffi | 0 | 1 | 0 | 1 | 2 | 2 | 2 | 0 | 2 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 0 |
| puncticollis | 0 | 1 | 0 | 0 | 2 | 2 | 2 | 0 | 2 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 0 |
| tikal | 0 | 1 | 0 | 2 | 2 | 1 | 2 | 0 | 1 | 2 | 0 | 1 | 1 | 1 | 0 | 2 | 0 | 1 | 1 | 0 |
| subtinctus | 0 | 1 | 0 | 2 | 2 | 2 | 2 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 2 | 0 | 1 | 1 | 0 |
| whiteheadi | 0 | 1 | 0 | 2 | 2 | 1 | 2 | 0 | 1 | 2 | 0 | 1 | 1 | 1 | ? | 3 | ? | ? | 1 | ? |
| g.gilvipes | 0 | 1 | 0 | 1 | 1 | 1 | 2 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | ? | 3 | ? | ? | 1 | 0 |
| g.peruanus | 0 | 1 | 0 | 1 | 1 | 1 | 2 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | ? | 3 | ? | ? | 1 | 0 |
| xingu | 1 | 1 | 0 | 1 | 1 | 1 | 2 | 0 | 1 | 6 | 0 | 1 | 1 | 1 | 1 | 2 | 0 | 1 | 0 | 1 |

Some other changes, such as adjusting the width of the taxon name column in the editor (see ["Changing the width of columns" on page 269](#)), will also be reflected in the pretty printed matrix.

You may wish to reduce the size of the matrix by setting the Scaling factor in the **Page Setup** dialog box to less than 100%.

Exporting matrices to be formatted in another program

An alternative method of printing the matrix is to use the **Simple Table** item of the **Export File** submenu of the **File** menu to create a text file of the data matrix that can be formatted in a word-processing, page-layout, or spreadsheet program (see ["Simple table files" on page 145](#)), or by editing and printing a copy of the NEXUS file itself from within a word-processing program.

Saving the pretty print matrix into a graphics file

If you wish to edit the pretty printed matrix in a graphics program, you will need to capture the image created while printing into a PostScript file. To create this PostScript file, follow the instructions in the section

["PostScript files" on page 467.](#)

We have not explored the use of MacClade-generated PostScript files in many graphics programs, but we have tested it with Canvas 7™. To open the PostScript file containing a pretty printed matrix in Canvas 7, drop the PostScript file onto the Canvas 7 icon. Canvas will present you with an EPSF Import Options dialog box. Choose "Create Canvas Objects" when importing. If the text looks too spaced out, you will have to select the entire image once opened, and choose **Normal** from **Kerning** submenu of the **Text** menu in Canvas.

Printing character and state names

The State Names and Symbols window can be printed when it is the frontmost window using the **Print Names** item in the **File** menu. The printout generated will be relatively crude, and will be similar to the contents of the window itself. If you wish a complete list of names for all characters, have the State Names and Symbols window as the frontmost window, and choose **Print All Names** from **File** menu.

Printing list windows

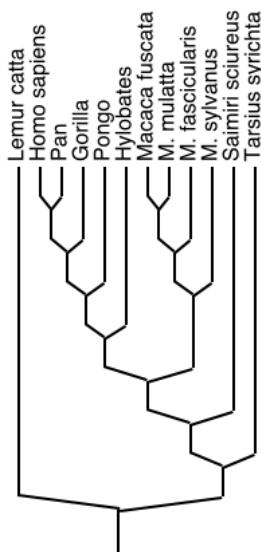
You can print any list window when it is frontmost on the screen by choosing **Print List** from the **File** menu. The list window will be printed more or less as it appears on the screen.

Printing text windows

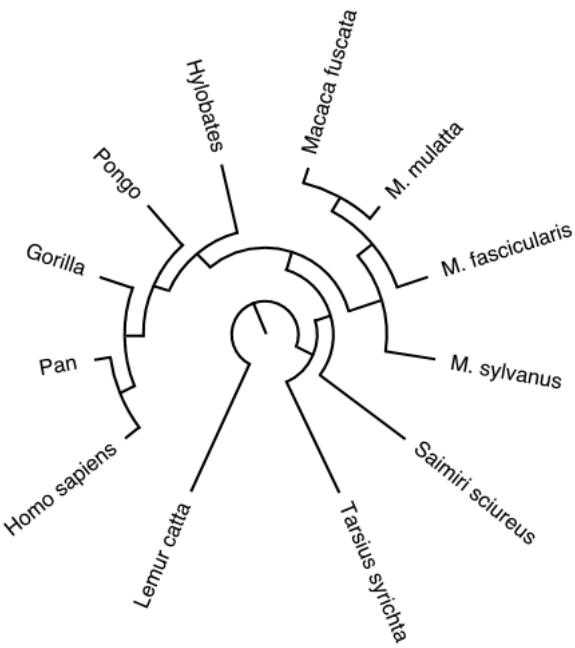
To print the contents of the File Notes or Notes about Trees windows, choose **Print Notes** from the **File** menu when the window is frontmost.

Printing trees

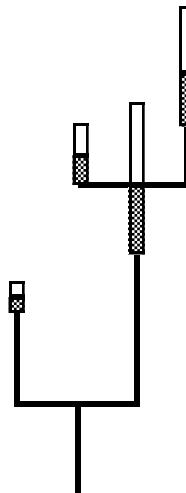
MacClade's tree printing facility allows you to print trees in a wide variety of forms. Characters can be traced or not, changes indicated, and branches may be slanting, square, or proportional to branch length. Taxon names may be written diagonally at varying angles and with varying fonts and styles. Five examples follow.



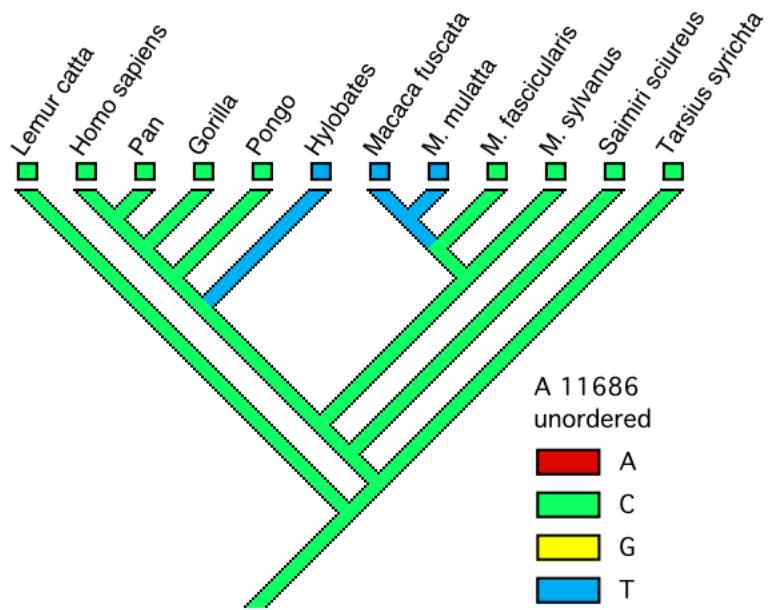
Example 1:
Partly angled branches



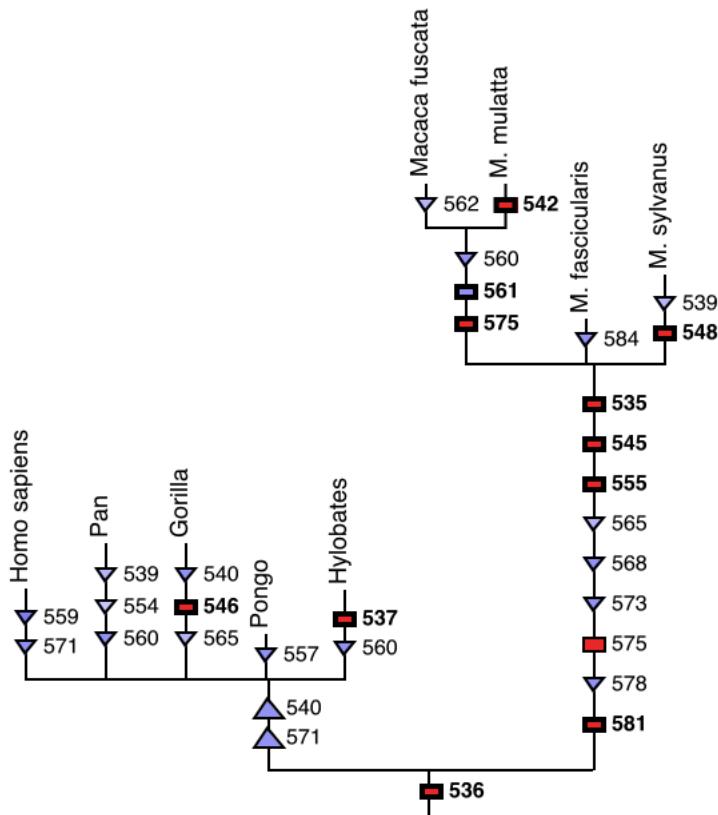
Example 2:
Circular tree



Example 3:
Branch lengths in
Trace All Changes



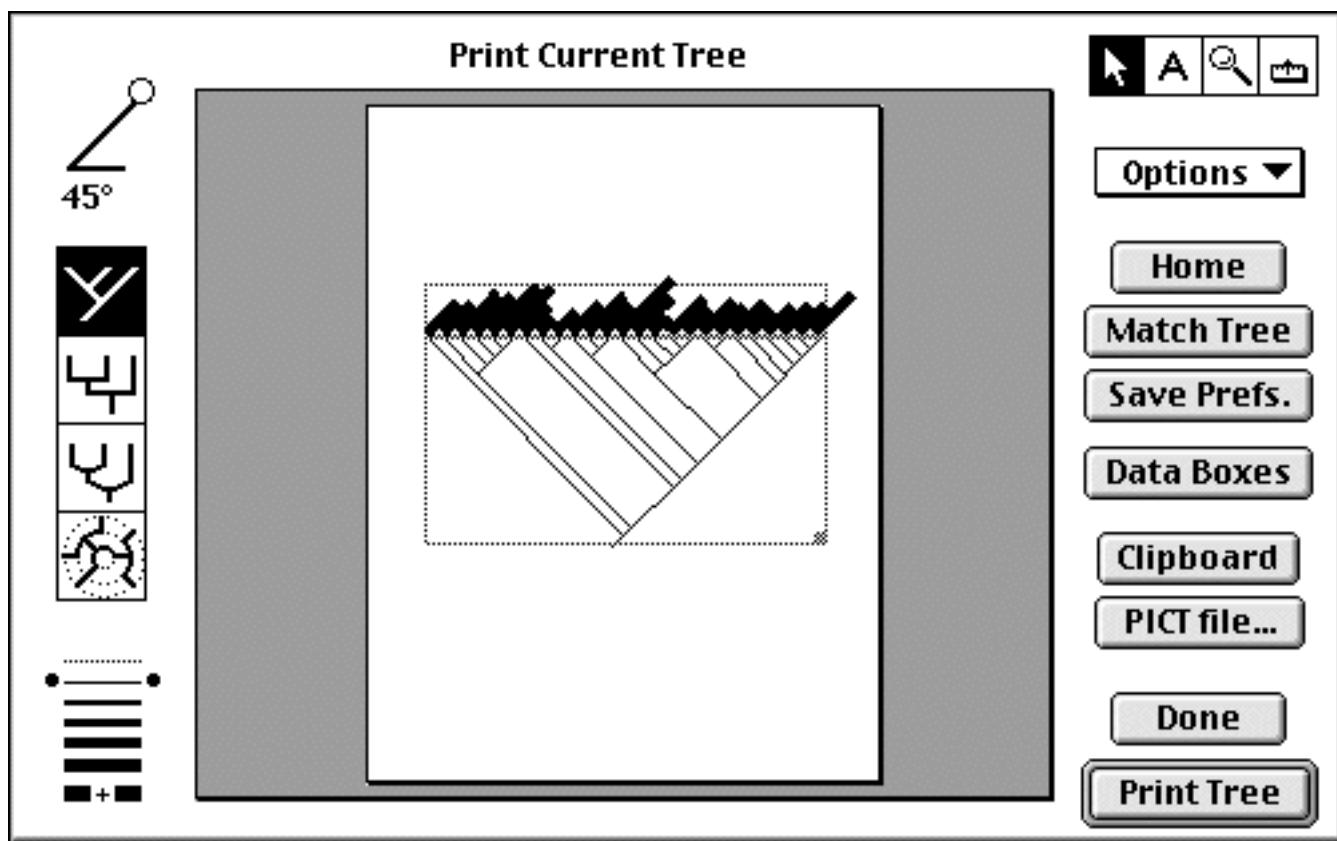
Example 4: A traced character



Example 5: Trace All Changes with bars indicating nature of character change elsewhere on tree

If you wish to print a traced character, you must trace it first in the tree window. To print branch lengths according to Trace All Changes, Trace All Changes must be displayed already in the tree window.

Trees can be printed when the tree window is on screen and it (or one of its accessory windows) is the front window; multiple trees can be printed if they are selected in the tree list window. Select **Print Tree**, in the **File** menu, and MacClade will present a dialog box that has an image of the tree on the page, as it will be printed, in the center:



The tree can be printed using the Print Tree button. The PICT file button will cause the tree image to be saved as a PICT graphics file, rather than to the printer. For a more complete discussion of graphics files, see "[Saving graphics files](#)", later in this chapter. If you click on the Clipboard button, then the image of the tree will be saved to the Clipboard for pasting into a graphics, word-processing, or other program.

If you click on the Data Boxes button, the Data Boxes (a graphical overview of the data matrix) will be printed.

Making the printed tree settings match the tree on the screen

Some display features, such as the shape of the tree, fonts of taxon names, and the manner of labeling traced branches, are controlled separately in the tree window and the **Print Tree** dialog box. For example, you can request angled branches for printing, and square branches for screen display. Other features, such as display of branch numbers, are controlled solely from within the tree window. In particular:

1. The controls are separate for the taxon name fonts, whether patterns, colors, or grays are used in tracing branches, and all options in the **Trace Labeling** dialog box for Trace All Changes except whether ambiguous changes are distinguished. Changing one of these options in the tree window will not affect values in the **Print Tree** dialog box.
2. Controls to change other options in the **Trace Labeling** dialog boxes are present in both windows, and changing values in one window affects the values in the other.

3. Controls for showing node numbers, showing evolve segments, adjusting the content of the taxon labels, and the patterns or colors used in tracing are available only in the tree window, not in the **Print Tree** dialog box.
4. Some options are relevant to only one of the windows, and only have controls in that window (e.g., branch lengths).

If the Print Tree dialog box is visible, and you wish to make the options for printing match the display of the tree in the tree window, use the Match Tree button. When pressed, this adjusts the display settings in the Print Tree dialog box to match (at least somewhat closely) the display settings in the main tree window.

Preview tools

At the top right of the dialog box is a palette of four tools — an arrow, an "A", a magnifying glass, and a ruler. These tools are used to edit the preview of the tree image, or to obtain information about it. The functions of the tools are as follows:

| Tool | Description |
|------|--|
| | The arrow is the basic tool that allows you to move boxes, resize the tree box, and so on. This tool will change to a hand when you move it over parts of the preview that you can move. |
| | The text tool allows you to change the font and style of text on the tree and legends. With the tool, touch on the portion whose text you want to change, and a menu will pop up. Choose the font, size, and style. |
| | If the magnifying-glass tool is touched on the image of the previewed tree, a close-up of the tree will be shown. Click again to return to the regular view. Another way to obtain a close-up view is to double-click on the tree. |
| | The ruler provides information about the positions of each of the boxes in the preview window. If you touch on a box with the ruler, the top left coordinates of the box will be written at the bottom of the dialog box. By recording these numbers, you can duplicate the exact positions of boxes the next time you print. You can see the size of the tree box by clicking on its grow box with the ruler. |

NOTE: Holding down the Option key will temporarily make the text tool the current tool, holding down the Command key (⌘) will make the arrow the current tool, and holding down the Shift key will make the ruler the current tool.

Adjusting tree shape

Trees can be printed with diagonal branches, with square branches, with an intermediate form in which the branches are slanted at the base but vertical near their tips, or in a circular fashion. To choose among

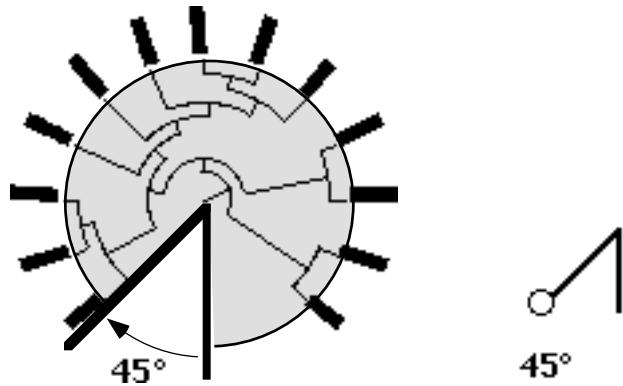
these forms, select the desired form in the palette on the left of the Print Tree dialog box:



Circular trees print much better on PostScript printers than on other printers. If you do ask to print circular trees on a non-PostScript printer, but have Angle Names (PostScript) checked, the taxon names will not print properly. Any feature of MacClade that entails tracing or labeling branches is not compatible with circular trees. There is currently a bug in circular tree printing that prevents MacClade from properly printing multiple circular trees per page on a PostScript printer.

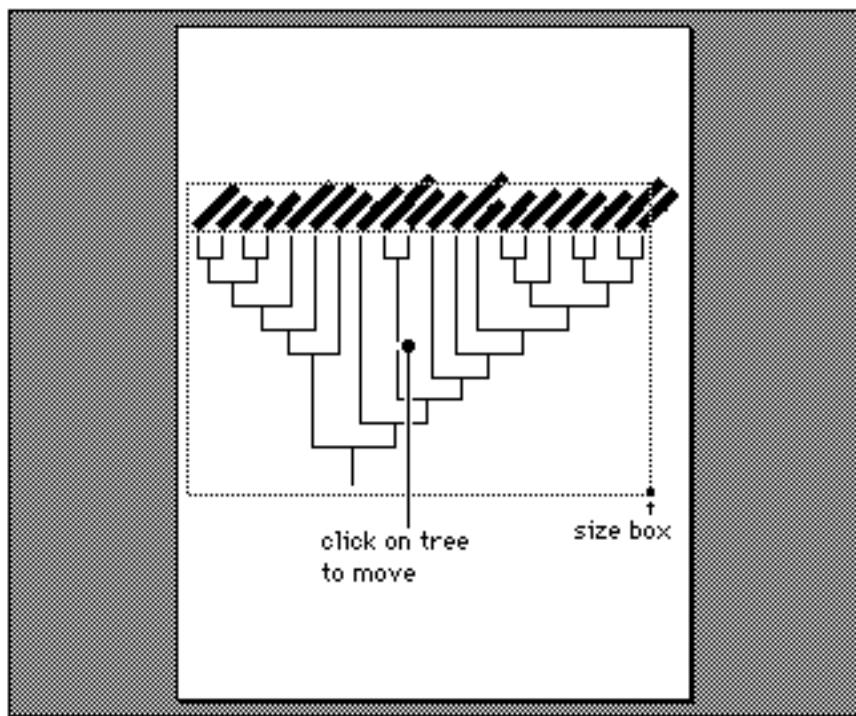
Angle of gap in circular trees

For circular trees, the angle of the gap at the bottom of the tree can be adjusted. For example, in the tree below, left, *half* the angle of the gap is 45°. To adjust the angle, grab and move the white ball on the end of the stick (in the upper-left corner of the dialog box), as shown below on the right.



Adjusting size and position of tree on page

The central window with the image of the tree shows a preview of how the tree will appear when printed. The central window looks something like this:



To move the tree on the page, click on the tree and drag it using the arrow tool (which will change to a hand when the pointer is over the tree). To adjust the width or height of the tree, click on the size box on the lower right-hand side of the tree, and drag it.

If you press the Home button, the tree will be returned to its default position and size on the page.

Drawing size

The **Drawing Size** item of the **Options** pop-up menu presents a dialog box, in which you can choose the number of pages the tree image occupies. In the dialog box the size of the selected area indicates the size of the image. For example, the selection shown below indicates an image 3 pages wide by 2 pages high.



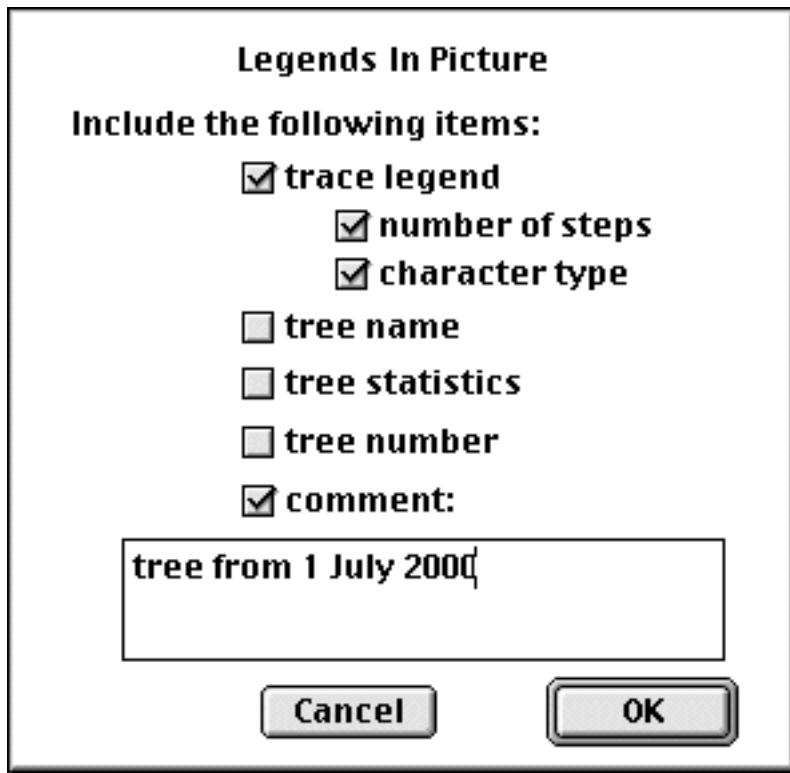
Multiple trees per page

The **Trees Per Page** item of the **Options** pop-up menu allows you to specify the number of trees to be printed horizontally and vertically on each page. This option is available only if the tree image occupies one page (see "Drawing Size", above).

Altering text and legends

You may see other boxes in the preview window (such as the legend for a traced character); you can move these around the page simply by dragging them.

These items can be removed or added from the page using the **Legends** dialog box from the **Options** pop-up menu. This dialog box allows you to choose whether the tree name, tree statistics (e.g., CI, treelength), trace legend, or comment are to be displayed.

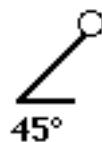


The comment is a piece of text you can add to the image. To add a comment, enter the text and check the "comment" box. If you wish to have the number of steps or character types included in the trace legend, then check those boxes.

The type font, size, and style of text on the image can be adjusted by clicking the text tool (**A**) on the portion of the image, and choosing the appropriate items from the pop-up menus that are presented.

Adjusting taxon names

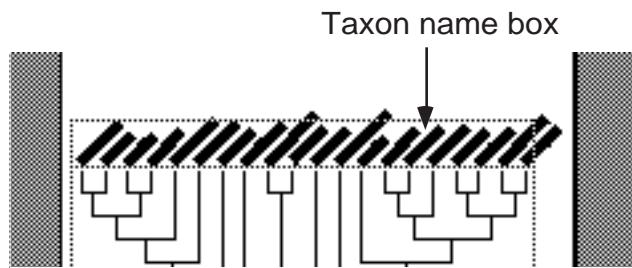
The angle of the taxon names for all trees except circular trees can be changed by grabbing and moving the white ball on the end of the stick in the upper-left corner of the dialog box:



This can be done only if the **Angled Names (PostScript)** menu item from the **Options** pop-up menu is checked. This will work on any PostScript printer, but is unlikely to work on non-PostScript printers. For non-PostScript printers, MacClade will in general print vertical taxon names. (We say "unlikely" and "in general" as some printer drivers for non-PostScript printers may provide some of the facilities of PostScript, including angled text in MacClade trees.) If you turn off the **Angled Names (PostScript)** option, then the names will always be printed vertically for noncircular trees.

For circular trees, you can ask MacClade either to print the names horizontally (by turning off **Angled Names**), or to print each name at the same angle as the branch to which it connects.

To change the font of the taxon names, touch the text tool (**A**) on the taxon name box immediately above the box surrounding the tree. A pop-up menu will drop down, from which you can choose the font, size, and style of the text.



Branch Appearance

Branch widths

Branch width can be varied by choosing one of the widths shown in the icon below.



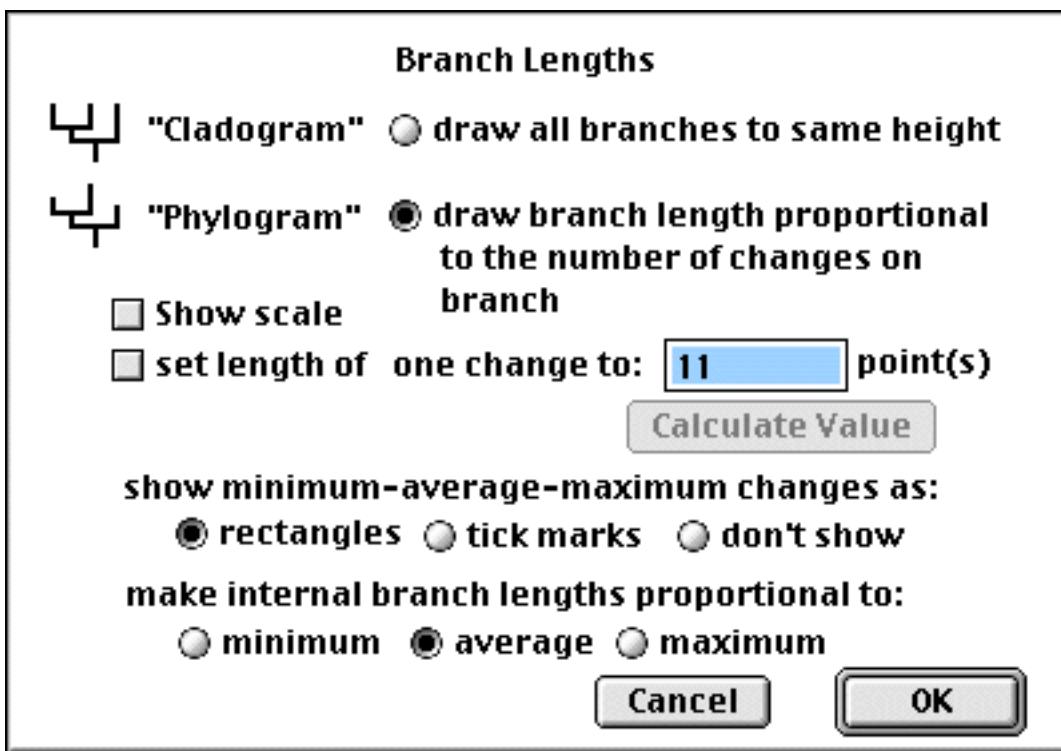
If the upper, dotted line is chosen, then the tree will be printed using very thin lines (1/288 of an inch) on PostScript printers, and single-point (1/72 of an inch) width lines on non-PostScript printers. If the bottom bar, with the "+", is chosen, a dialog box is presented allowing you to specify a width in points (one point is 1/72 of an inch).

Branch lengths

When Trace All Changes is active on the current tree, and trees are square, MacClade can print trees that have branches longer or shorter to show the amount of reconstructed change on each branch; the length of the branch is drawn proportional to the amount of change. The amount of change is either the number of changes, or the weighted length that the branch contributes to treelength. The branch length is taken

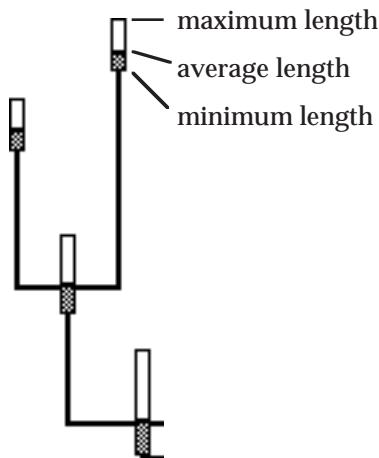
directly from the number of changes or length shown in Trace All Changes. Branch lengths cannot be shown on soft polytomies (and thus the branches involved in a polytomy will always be shown of zero length; see [page 369](#)).

To adjust how MacClade displays branch lengths, choose the **Branch Lengths** item of the **Options** pop-up menu, and the following dialog box will be presented:

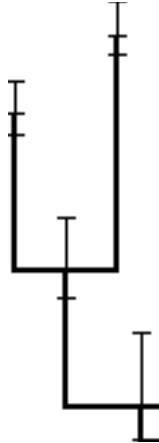


In the dialog box, you can alter the scale of the branches, and have a scale bar printed on the right side of the tree, if you wish.

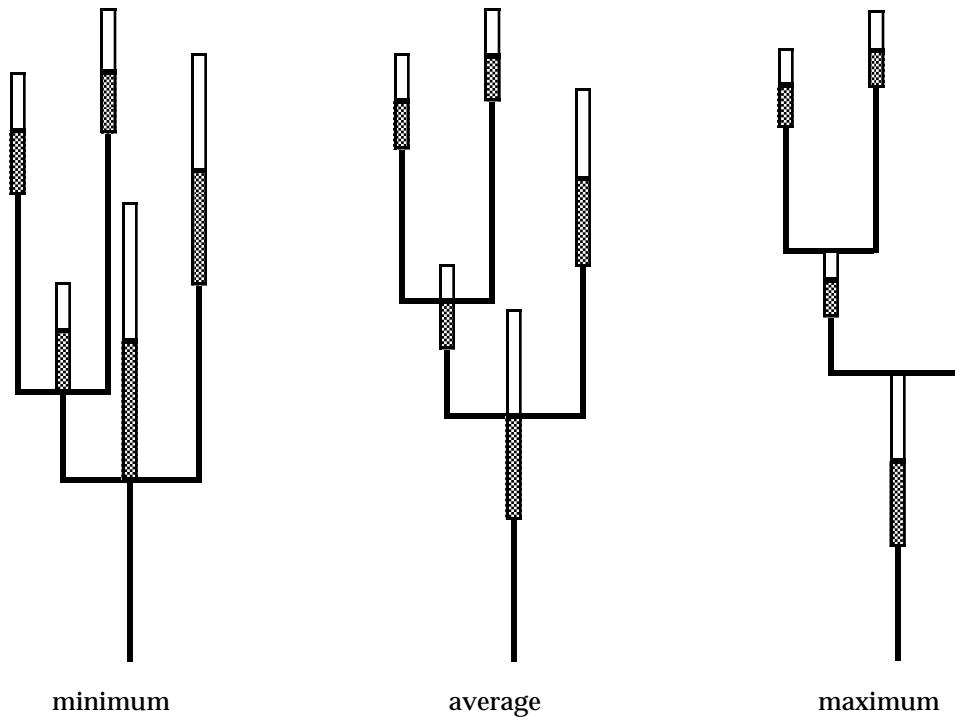
If minimum-average-maximum changes are calculated, then all three will be visible on each branch, with gray and white bars indicating the minimum, average, and maximum lengths over all equally parsimonious reconstructions of character evolution:



If you prefer, the display can be altered to use tick marks rather than rectangles:



By default, the length of internal branches is proportional to the average branch length. If you prefer, you can adjust the length to match the minimum or maximum branch lengths:



Trace labeling

When Trace All Changes is displayed or a character is traced in the tree, the **Trace Labeling** item in the **Options** pop-up menu allows you to choose how the trace is to be displayed. For full details on the options, see [page 360](#) if Trace All Changes is shown, or [page 350](#) if a character is traced on the tree. If Trace All Changes is shown, and minimum-average-maximum changes are calculated, then you cannot trace the changes using a pattern or color if you also ask MacClade to draw the length of branches proportional to the number of changes.

Branch shading

The **Branch Shading** submenu of the **Options** pop-up menu allows you to choose whether patterns, colors, or grays are used in tracing characters or Trace All Changes on the tree.

Printing multiple trees

To print multiple trees, open the tree list window ([Chapter 10](#)), and select the trees ([page 163](#)) you wish to print. Choose **Print Multiple Trees** from the **File** menu, and you will presented with the Print Tree dialog box, in which you can choose the options for the trees to be printed.

Print trace for all characters

If the **Print All Characters** item in the **Options** pop-up menu is activated when a character is traced in the tree window, then a tracing for each included character will be printed. You may want to print multiple trees per page so as not to use too much paper.

Printing all MPRs of a traced character

If the **Print All MPRs** item in the **Options** pop-up menu is activated, all equally parsimonious reconstructions of evolution of the traced character are printed in sequence. You may want to print multiple trees per page so as not to use too much paper. Remember, there can be many MPRs.

Printing charts

Charts can be printed when they are on the screen by choosing **Print Chart** in the **File** menu when the chart window is frontmost.

If you wish to have the chart fill the printed page, turn on **Full Page Print** in the **Chart** menu. If **Full Page Print** is unchecked, MacClade will print a chart of the size seen on the screen.

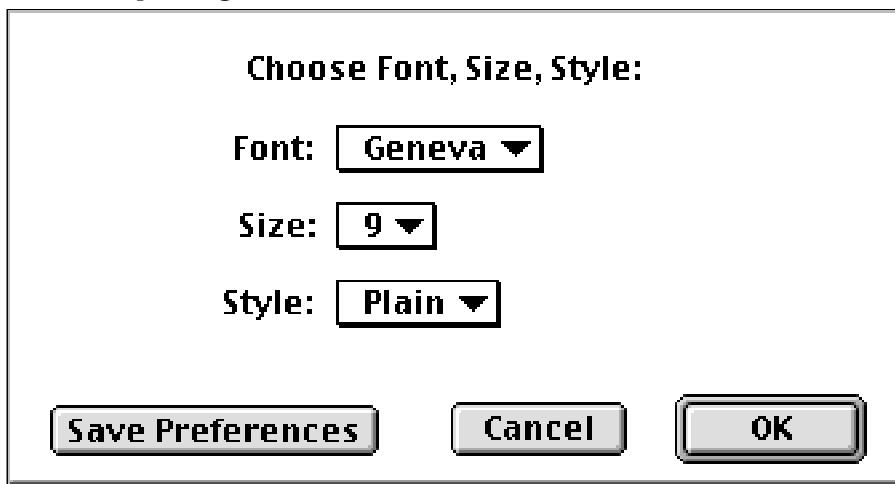
Printing other components

A number of other items (general notes, branch lists, footnotes, pictures, data boxes, user-defined types, genetic code, and continuous data) can be printed using the **Print Other** submenu in the **File** menu. These items all share choice of fonts through a dialog box presented when you request that they be printed.

Choosing fonts for printing

For printing many of the windows in MacClade, including the data editor, chart window, and list windows from within MacClade, the font used is the font visible on the screen. For printing the tree, control over fonts is provided in the **Print Tree** dialog box.

For the items in the **Print Other** submenu, however, MacClade gives you a choice of fonts to use when you request printing, in the dialog box pictured below. Choose the font, size, and style of the text to be printed, and press OK to continue printing.



Printing general notes about the file

By choosing **General Notes** from the **Print Other** submenu, MacClade will print a sheet with the current time and date, version of MacClade, and the status of some options:

```
MacClade version 4.0
Saturday, July 1, 2000: 9:02 PM

General Notes
=====
Data file: Hamamelidae

28 taxa

CHARACTERS
54 characters
54 characters included
No characters excluded
Character Transformation types:
    Types currently in use: unordered
    Directed types in use: no
    Current weight set stored as Equal weights
    Current character inclusion set stored as All included

Missing data, entire matrix: 191 cell(s), 12.63%
Gaps, entire matrix: 0 cell(s), 0.0%

TREES
Current tree repository: external tree file (Ham.island4)
Polytomies assumed to represent uncertain resolution
Current tree:
    Includes all 28 taxa in data file
```

Upon your request to print these general notes, MacClade will present you with a dialog box that allows you to choose the font of the printed notes.

Other components of the data file, such as the user-defined transformation type definitions, the footnotes, pictures, a list of character and state names, the current genetic code, Data Boxes, and continuous data can

be printed using the **Print Other** submenu in the **File** menu. Upon your request to print any of these, MacClade will present you with a dialog box that allows you to choose the font of the printed information.

Printing information for each branch

The **Branch Lists** or **Node Lists** item in the **Print Other** submenu prints a list of information about each of the branches. When a character is traced, whether discrete or continuous, the printout will indicate the state(s) reconstructed at the node above each branch. When **Trace All States** is displayed, a list of the states at the node above each branch is printed, or the frequencies of states, depending on the options selected in the branch list tool's pop-up menu (["More information about the changes occurring on branch" on page 367](#)). When **Trace All Changes** is displayed, a list of changes along each branch is printed. Branches are referred to by their numbers, which will be printed on the tree if **Show Branch Numbers** is chosen from the **Display** menu.

Printing all footnotes

This option prints all of the footnotes stored in the file.

Printing all pictures attached to cells

This option prints all of the pictures attached to cells in the data matrix.

Printing Data Boxes

If you ask to print the Data Boxes from the **Print Other** submenu, then the taxa will appear in the Data Boxes in the order of appearance in the data matrix, not their order in the current tree. To have Data Boxes printed in the order of appearance in the current tree, you must print Data Boxes from within the **Print Tree** dialog box in the **File** menu.

Printing user-defined types

This option prints all user-defined types.

Printing the genetic code

This option prints the current genetic code.

Printing continuous data

This option prints all continuous characters.

Saving graphics files

PICT files

As noted already in the discussion of tree printing, output that might normally go to the printer can instead be saved to graphics files of PICT format for later editing in a graphics program. This can be done for the tree window, chart window, and other windows using the **Save Graphics File** command in the **File** menu.

Ideally, you could then go into a graphics program that understands PICT files, open the graphics file saved by MacClade, and edit the image to your heart's content. Unfortunately, different graphics programs treat PICT files in different ways.

Most current programs deal correctly with rotated text in the tree (if you have **Angle Names (PostScript)**

turned on), but some may ignore the rotation, or deal with it improperly. If **Angle Names (PostScript)** is turned off, all of the programs should understand the rotated text, but it is of poorer quality.

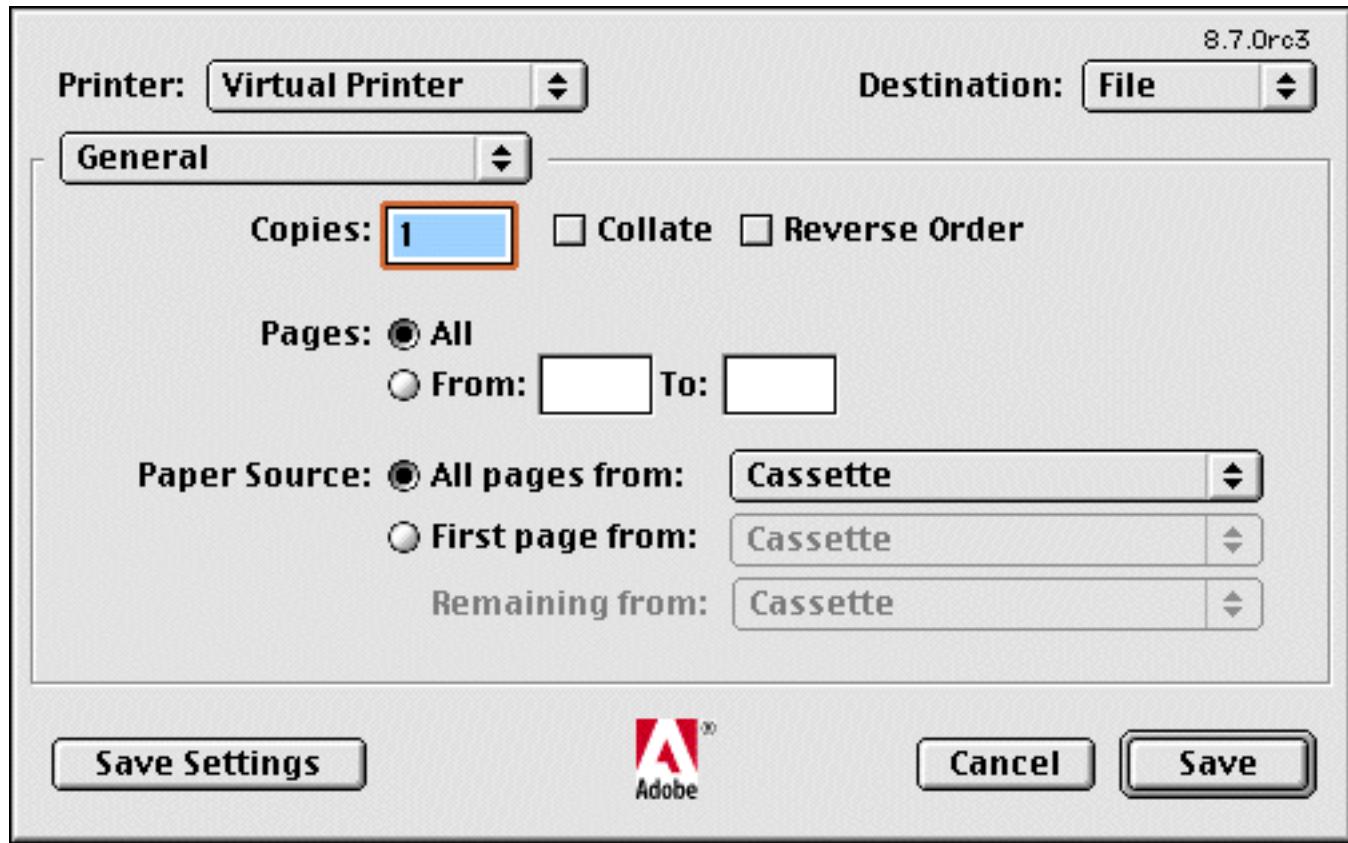
Branches in trees are more problematic. Although old versions of some programs (e.g., Canvas 3.5) correctly interpreted branches, newer versions (e.g., Canvas 7) do not, incorrectly presenting PICT files saved by MacClade. We have yet to find a current version of a graphics program that correctly interprets PICT files; you may wish to check the MacClade Web site (see [page 20](#)) for any news about this. (However, note that PostScript files are better interpreted; see next section.)

You may want to save the PICT file so that the Macintosh thinks it was created by MacDraw or some other graphics program, so that it will be easier to open directly from the Finder. To do this, set the creator of saved PICT files using the **Other Options** dialog box in the **Options for Saving** submenu of the **File** menu. To use this feature safely, you need to understand the concept of file creator and to know what the appropriate four-symbol code is. (For example, the four-letter creator code for Canvas 7 is DAD7; for Illustrator 8 it is ART5; for AppleWorks 5 it is BOBO.)

PostScript files

Rather than generating PICT files, you may find that generation of PostScript files will give better results, especially for trees.

To generate a PostScript file, make sure the MacOS is set to print to a PostScript printer, and then ask MacClade to print whatever you wish to save as a PostScript file. In the dialog box presented, choose "File" as the destination rather than "Printer". An example Print Job dialog box is shown below.



In choosing the destination to be a file, the PostScript image will be sent to a file on disk, rather than to a printer.

If you do not have a PostScript printer, and thus cannot set the MacOS to print to a PostScript printer, you can instead install AdobePS, a free PostScript printer driver that lets you print PostScript images to a file. It is available from Adobe's Web site, <http://www.adobe.com/>.

Trees saved as PostScript files and then imported into Adobe Illustrator 8 are interpreted correctly, with one minor exception: equivocal and other patterned branches are shown in gray. Nonetheless, this is the best solution we have found to editing of tree images produced by MacClade in graphics programs.

Saving text files

The **Save Text File** submenu in MacClade provides a means to save information to disk as text files. You can save a text file version of the General Notes, described above under "[Printing general notes about the file](#)" on page 465. You can save a list of the states reconstructed at the nodes of the tree, or changes reconstructed along the branches, described above under "[Printing information for each branch](#)" on page 466.

You can also save a text version of the character list window, by choosing **Save List as Text** from the **Save Text File** submenu, which will produce a file somewhat like this:

```
MacClade version 4.0
Monday, 10 July 2000: 9:11 AM

Data file: Bembidion.

Current tree: 1 of 4 common.

      Character          Type    Weight   States   Steps
1. silver spots     • ordered      1        2        1
2. interval 3 micro. • ordered      1        4       10
3. interval 4+5 micro. • ordered      1        4       20
4. interval 6+7 micro. • ordered      1        4       14
5. outer mirror     • ordered      1        2        3
```

A chart can be saved as a text file for later editing and examination. For all charts, **Save Chart As Text** can be called from the **Save Text File** submenu of the **File** menu to save a textual version of the chart.

An example chart is:

```
MacClade version 4.0
Monday, 10 July 2000: 9:15 AM

Changes in all characters in Tree

      mean
numbers
 0      4      5      2
 5      0      8      6
 2      1      0      0
 4      6      7      0

frequencies, whole matrix normed
 0.0      0.080    0.100    0.040
 0.100    0.0      0.160    0.120
 0.040    0.020    0.0      0.0
 0.080    0.120    0.140    0.0

frequencies, rows normed
 0.0      0.364    0.455    0.182
 0.263    0.0      0.421    0.316
 0.667    0.333    0.0      0.0
 0.235    0.353    0.412    0.0
```

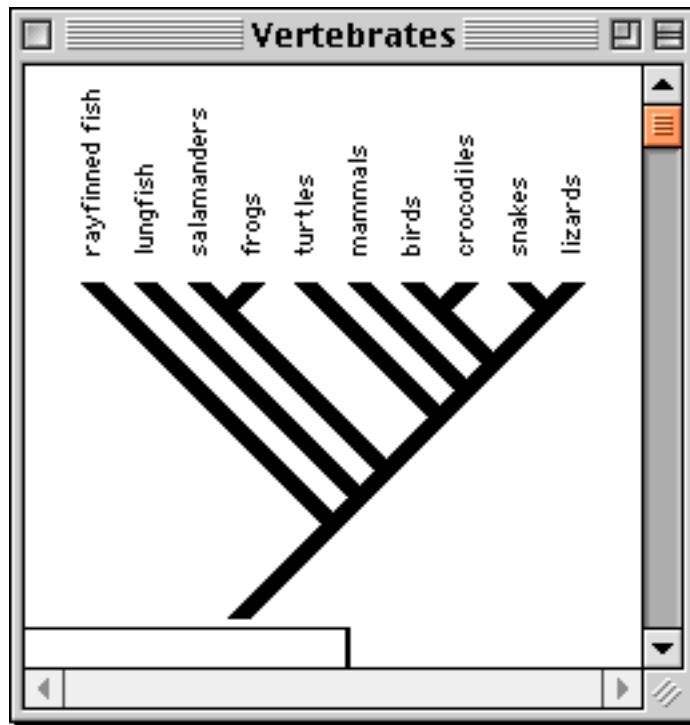
As well, if the Character Steps/etc. or State Changes & Stasis chart is requested, the "log file" check box can be selected in the **Chart Options** dialog box before the chart is calculated, and a detailed text file will be saved. This is described in ["Text files with values for each tree" on page 385](#).

If the columns in the text file do not align, you may need to adjust tabs and switch to a font like Courier or Monaco in your word processor.

Text files can be saved in various other places in MacClade, such as a report on random tree generation, as described elsewhere in this manual.

Placing a text version of the tree into the Clipboard

If you wish to copy a text version of the current tree in the tree window, perhaps to include in an email message, hold down the Option key and choose **Copy Text Tree** from the **Edit** menu. For example, if the tree window displays the following tree:



then **Copy Text Tree** will place a text version of that tree into the MacOS Clipboard; the tree will look as follows, if pasted into a program and displayed using a nonproportionally spaced font (e.g., Courier):

```
===== rayfinned fish
   |
   ===== lungfish
   |
   ===| ===== salamanders
   |
   ===| ===== frogs
   |
   ===| ===== turtles
   |
   ===| ===== mammals
   |
   ===| ===| birds
   |
   ===| ===| ===| crocodiles
   |
   ===| ===| ===| snakes
   |
   ===| ===| ===| lizards
```



USING MACCLADE AND PAUP* TOGETHER

26

In developing MacClade we have worked closely with David Swofford, the developer of PAUP* (Swofford, 2000), to ensure that the two programs are compatible and complementary. The two programs are often used together: MacClade to edit files, to set up constraint trees and assumptions, PAUP* to do automatic searches for optimal trees, and MacClade to explore the trees further and analyze character evolution upon them. In this chapter we discuss some of the issues in using the two programs together. Among the most important are the incompatibilities and inconsistencies that still exist between PAUP* and MacClade, most of which concern differences in features supported by the two programs.

If you use PAUP* and MacClade, we recommend that you acquire the latest version of both programs, as they will be most compatible with one another. MacClade's data files will be more acceptable to later versions of PAUP than to earlier versions. The description of incompatibilities between MacClade and PAUP* apply to versions 4.0 of both programs. PAUP* changes relatively rapidly; some of the incompatibilities may be eliminated in the near future. Check MacClade's Web pages (see [page 20](#)) for the latest news on any changes to the incompatibilities described herein.

A scenario of using MacClade and PAUP*

Here we present a typical scenario of using MacClade and PAUP* together. Of course, in your own work you would be unlikely to perform the exact same actions in the same order, but we present this example to indicate how the programs can interact.

In MacClade, open a new file and enter a data matrix and some user-defined types. Apply weights and types to some of the characters. Then save the file as "My genus". Start PAUP*, and open and execute this file. The assumptions (such as weightings) in effect when you saved MacClade will be in effect in PAUP* when the file is executed. Now do a branch-and-bound search to find parsimonious trees, and use **Save Trees to File** to save these trees to a tree file named "My genus.trees". Go back to MacClade, and go to its tree window, asking for the tree file you just saved in PAUP*. Scan through the trees to see how they look, tracing the evolution of particular characters of interest. Now close the tree file and rearrange the tree on screen to match a previously published tree. Store the tree with the name "published". Save the file. Now go to the chart **Compare 2 Tree files** and select the data file (containing the published tree) and "My genus.trees" (containing the PAUP* trees) to see if the published tree has consistent differences with PAUP*'s trees. Now build a new tree that is mostly unresolved but that indicates as monophyletic one special group that was not monophyletic in PAUP*'s trees. With that tree in MacClade's tree window, choose **Constraint Tree** from the **Place PAUP Command** submenu of the **Edit** menu; this will store the constraint tree in a PAUP block. Store the tree with the name "constraint". Save the data file. Go back to PAUP*, *close and reread the data file*. The constraint tree will now be processed by PAUP*. Do another tree search, this time enforcing the topological constraints tree "constraint". This will find the shortest trees that have this special group monophyletic. After saving the resulting trees, go back to MacClade and analyze evolution of important characters on the various trees, summarizing the reconstructed character evolution using the charts.

Using both programs simultaneously

In the scenario described above there were a number of steps of saving files that may have seemed super-

fluous. They were not, even if you are using MacClade and PAUP* simultaneously. MacClade reads a data file from disk and operates on it while it is in the computer's RAM memory. Any changes you make to the data file are not written to the disk until you give the **Save File** or **Save File As** command. If you are editing a data file in MacClade and want to move to PAUP* and work on the same data file, remember to save the data file to disk before moving from MacClade to PAUP*, where you must then reread the data file from the disk. Otherwise, PAUP* won't know about the most recent changes to the data file. Because PAUP* does not have access to MacClade's RAM copy of the data file, it must access the data file via the disk. This principle works in reverse — if PAUP* has done a tree search, don't expect MacClade to have access to the trees stored in PAUP*'s memory. You must save the trees to a file on disk first, after which MacClade can read the trees from the file. Always remember that the files *as they are saved on the disk* are the means by which all information is passed between MacClade and PAUP*.

Control over NEXUS blocks

You can edit NEXUS blocks directly in MacClade. This can be useful for adding commands to blocks that MacClade does process, or for creating entirely new NEXUS blocks. For example, you can add a CHAR-PARTITION command to the SETS block, which will then be available to PAUP* (see "[Adding foreign commands to a processed NEXUS block](#)" on page 131).

Embedding PAUP* commands in data files

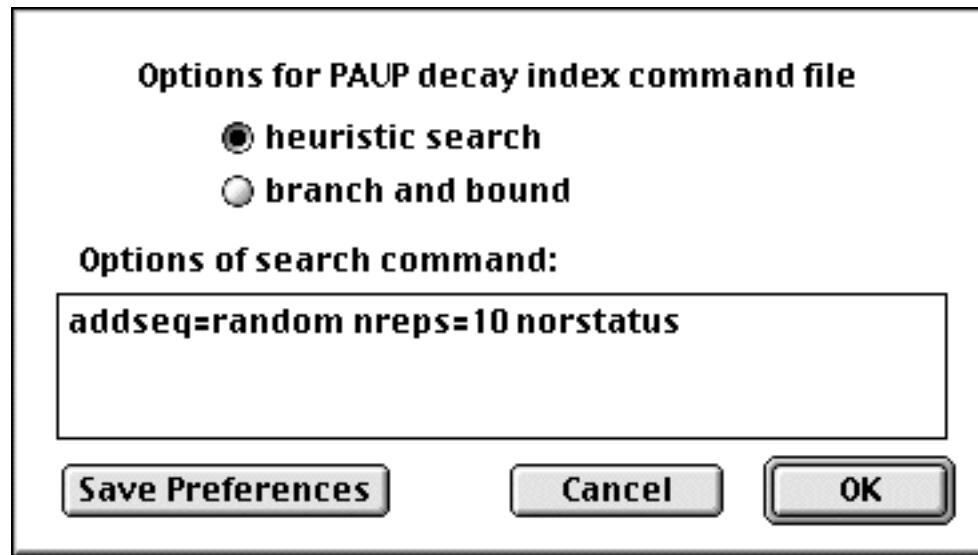
MacClade allows you to manipulate PAUP blocks, which allows you to embed commands into your file that PAUP* will process upon execution. Some of the features of MacClade for manipulating NEXUS blocks are specifically designed for PAUP*; see "[Special features for PAUP blocks](#)" on page 134. In particular, you can:

1. Turn the tree in MacClade's tree window into a constraint tree in a PAUP block (see "[Constraint trees](#)" on page 134).
2. Specify outgroups (see "[Defining outgroups](#)" on page 135).
3. Define two commonly used PAUP* commands, which you can then add quickly to any PAUP block (see "[Custom commands](#)" on page 136).

Calculating decay index values

MacClade's **Decay Index PAUP File** in the Σ menu allows you to generate a PAUP* command file that can be used to calculate decay indices (also called Bremer support; Bremer, 1988; Donoghue et al., 1992) of the clades in a tree.

To calculate decay indices for all clades shown on the current tree, choose **Decay Index PAUP File**. The dialog box presented lets you choose the nature of the search that will be conducted by PAUP* to calculate decay indices:



Choose the nature of the search (heuristic or branch and bound) and include any options of the search command in the specified box. If you do not specify any options for the search command, PAUP* will conduct a default heuristic search or branch and bound. After pressing OK you will be presented with a dialog box for choosing the file name and location of the command file produced. MacClade will then create a text file of PAUP* commands. If you execute the data file in PAUP*, ask to log the output to disk, then execute the PAUP Decay Commands file that MacClade created, PAUP* will then conduct a series of searches. Examination of the output stored in the log file will allow you to calculate the decay indices.



For example, imagine you wish to find the decay indices of the clades in the most parsimonious trees for the "Primate mtDNA" matrix (in the Molecular folder in the Examples folder). The following steps would allow you to do this:

1. Find the most parsimonious trees in PAUP*. There should be two trees of length 1153.
2. Compute the strict consensus trees of these two trees in PAUP*, in doing so, choose the "Output to treefile" option so that the strict consensus tree is saved in a tree file.
3. In MacClade, open up the "Primate mtDNA" file, go to the tree window, and open up the tree file containing the consensus tree. Display the consensus tree on the screen.
4. In MacClade, choose **Decay Index PAUP File** from Σ menu. Choose heuristic search as the search method (you may wish to add options to make it a more thorough search). Press OK. When you are asked for the name of the file to be saved, call it the default PAUP Decay Commands, and save it in a place you will remember.
5. Go back into PAUP*, and (making sure the "Primate mtDNA" file is still the active file), and turn off **Log Output to Disk** from the **File** menu of PAUP*.
6. Open and Execute the PAUP Decay Commands file in PAUP*. PAUP* will conduct multiple searches, one for each clade in the consensus tree, and it will save the results in the log file.

7. Once it is done, open the log file generated, and examine it. You will see multiple sections similar to the one that follows:

```
===== clade1 =====

Constraint-tree "clade1":

/----- Pongo
+----- Hylobates
+----- Macaca fuscata
+----- M. mulatta
+----- M. fascicularis
+----- M. sylvanus
+----- Saimiri sciureus
+----- Tarsius syrichta
+----- Lemur catta
| /----- Homo sapiens
\-----+----- Pan
\----- Gorilla

Of the 13,749,310,575 distinct rooted binary trees for 12 taxa, 103,378,275 are
compatible with this constraint tree

Of the 654,729,075 distinct unrooted binary trees for 12 taxa, 6,081,075 are compatible
with this constraint tree

Heuristic search settings:
[setting omitted]

Heuristic search completed
Total number of rearrangements tried = 521
Score of best tree(s) found = 1172
Number of trees retained = 1
Time used = 0.07 sec

1 tree saved to file "clade1.tree"
```

8. This output allows one to calculate the decay index of "clade1", which is the clade containing Homo, Pan, and Gorilla. The decay index of that clade is the score of the best tree found in this search (1172) minus the score of the most parsimonious trees (1153); the decay index is thus 19 for that clade.

Incompatibilities between MacClade and PAUP*

The programs are not fully compatible. That is, there are slight differences in defaults or in the ways that certain calculations are done, and there are some differences in the data file formats used by the two programs. The differences in data files concern different capabilities of the programs: thus, for instance, MacClade does not support the interpretation of gaps as extra character states, whereas PAUP* does not support the stratigraphic character type.

If you find an incompatibility other than those described below, especially in calculations, it may represent a bug in one of the programs. Reread the instructions for both programs to see if the incompatibility is an intentional difference in the features of the programs. If it appears to be a bug, please report it to the authors of the programs.

Data

PAUP 3 required that all multistate entries in the data matrix be uncertainties, or all be polymorphisms; mixtures are not allowed. PAUP*4 accepts matrices in which some cells represent partial uncertainties, and others are polymorphisms. If you do type in some cells as 0/1 in MacClade, and others as 0&1, PAUP*4 will default to assuming all are 0/1, unless you ask to treat them as different in the Parsimony Settings dialog box (see PAUP* manual). The difference between uncertainty and polymorphism does not make an important difference in finding parsimonious trees for unordered and ordered characters, but could affect what are the parsimonious trees for other character types such as irreversible, Dollo, and step-matrix types.

Assumptions

MacClade does not support GAPMODE = NEWSTATE. That is, in MacClade gaps are treated as missing data.

PAUP*'s Ignore Uninformative and MacClade's **Exclude Uninformative** may not ignore the same characters, for several reasons. First, PAUP* has algorithms to calculate what might be uninformative for characters of unordered, ordered, irreversible, and Dollo types; MacClade finds uninformative characters only when they are unordered or ordered. Second, when some terminal taxa are polymorphic, PAUP* may fail to exclude some characters that are actually uninformative. Third, MacClade's **Exclude Uninformative** excludes characters that are uninformative over the set of taxa *in the current tree*, which may be only a subset of the file's taxa.

PAUP 3 does not support weights with decimals. You should convert weights to integral values for use with PAUP 3. PAUP*4 does accept decimal weights.

User-defined types

MacClade cannot handle step matrices with values above 999; PAUP* allows higher ones. When MacClade finds a matrix with higher values, it divides the matrix by 10s until all values are under the limit. This affects steps counted and possibly the character reconstruction.

MacClade and PAUP* differ in how they treat terminal taxa polymorphic for three or more states in a user-defined type character. MacClade and PAUP* also differ in how they treat unnamed states in character state trees. These differences are described under ["Treelength"](#).

PAUP* does not support the stratigraphic type. If you use the stratigraphic type in a data file, earlier versions of PAUP* will give a warning about the illegality of "STRATTYPE" or "strat" and stop processing the data file; later versions will give a warning and convert the assignment of any stratigraphic characters to the default type.

PAUP 3 does not support user-defined step matrices that contain numbers with decimal points (REAL-TYPES). Convert these decimal values to integers for use in PAUP 3. PAUP*4 does accept decimal values in step matrices.

Trees

MacClade can store trees in a data file or tree file with different numbers of taxa. For example, some of the trees might contain taxa 1–7, others 4–9. PAUP*, however, on reading the file, automatically adds taxa to trees until they contain all currently available taxa (either all taxa in the data matrix or fewer if you have explicitly told PAUP* to delete taxa).

MacClade can treat polytomies as uncertainty of resolution ("soft" polytomies) or multiple speciation ("hard" polytomies); PAUP* treats polytomies always as hard. This can generate differences in treelengths and character reconstruction.

MacClade treats all trees as rooted trees, as required by a number of MacClade's features. The State Changes & Stasis chart, for instance, needs a root to determine whether a change was a 0 to 1 or a 1 to 0 change. Users will notice that PAUP*, on reading a MacClade file with all trees saved as TREEs, requests to deroot these trees whenever no directed transformation types are in use. If you ask MacClade to write trees as UTREEs, or if you allow PAUP* to deroot the trees, you should be aware that the rooting as shown by MacClade may be lost in the process, and that PAUP* may substitute, in a sense, an alternative root. PAUP*'s unrooted trees are not purely unrooted, for something like a root is arbitrarily chosen to be between the first taxon in the matrix and the rest (if no outgroups are designated). This "root" is used both to polarize PAUP*'s ACCTRAN/DELTRAN reconstructions and to orient branch swapping. It also may become the true root if directed transformation types are reapplied and the trees are rerooted.

These differences between PAUP* and MacClade tree rooting can cause confusion. Under the following section, we describe how one can, by mistakenly rerooting trees, obtain different treelengths under MacClade and PAUP*. One can also obtain unexpected results in interpreting character evolution. For instance, if you have a tree $(0,(0,((1,1),(1,0))))$ with a single unordered character, MacClade will show ambiguity as to whether there was a gain and loss or two independent gains. If you read the file into PAUP*, you get the request to deroot the trees. If you say yes and then in the **Show Reconstructions** dialog box ask for character changes, PAUP* will indicate a switch from 0 to 1 to 0 if the first taxon in the data matrix happened to be one of the first two taxa in the tree above. If you then go to MacClade's editor and move rows around to put one of the taxa with state 1 into the first row, save the file, and move back into PAUP* and reread the file (not having changed the tree or the data, only the sequence of taxa in the matrix), PAUP* will now indicate a switch from 1 to 0 to 1. PAUP* indicates different character changes before and after the moving of rows because (1) PAUP* uses as default reconstruction for its "character change" display the ACCTRAN reconstruction, and the ACCTRAN reconstruction depends on some sort of root, and (2) PAUP* chooses the first taxon in the matrix as the implicit "root" for purposes of ACCTRAN when rooting is not specified otherwise. PAUP* also uses an implicit root when presenting lists of changes and apomorphies, which may not be accurate if PAUP*'s implicit root does not match the explicit root chosen in MacClade.

Treelength

Users moving between PAUP* and MacClade may notice situations in which the two programs give a different treelength. Some possible causes are as follows:

1. You may have been using different versions of the data file. See the comments above ([page 471](#)) regarding the caution needed when using MacClade and PAUP* simultaneously.
2. MacClade and PAUP* may be using different sets of assumptions because of the different commands and parts of the data file supported by the two programs. MacClade does not read the PAUP block where you may have specified assumptions such as which characters are excluded. When you change assumptions in PAUP* using menu items (for instance, using **Set Character Types**), these assumptions are not saved with the data file when you save it. They are only ephemeral changes unless you ask PAUP* to save the assumptions to a separate file (which MacClade will not process). The only way to ensure that such changed assumptions are saved to the data file is to place them by hand into the ASSUMPTIONS block of the file in PAUP*'s editor, or specify them in MacClade and save the file (MacClade automatically writes the newly changed assumptions into the file).
3. MacClade allows integral values in step matrices to be only as high as 999; PAUP* allows values up to 32767. When MacClade encounters higher values, it will divide all values in the step matrix by 10s until the highest value is within MacClade's limit. This may result in MacClade's reporting not only a lower number of steps for the character but also accuracy will be lost as MacClade reduces the number of digits to 3. If only a subset of characters are assigned a user-defined type

that was so reduced, these characters will effectively have their relative weights reduced by a factor of 10 or more.

4. Unnamed states in character state trees cannot be assigned as states at ancestral nodes in the tree by MacClade, in contrast to PAUP*. This means MacClade may not get as low treelengths as will PAUP*. If it were important to allow unnamed states to be assigned in MacClade, you could manually create a step matrix that corresponds to the costs implied by the character state tree and that uses named states for the formerly unnamed states.
5. MacClade does not support the PAUP* character types `dollo.dn` and `irrev.dn`; MacClade will convert characters of these types to `dollo.up` and `irrev.up`, which may affect the treelength.
6. MacClade always treats gaps like missing data for calculations; PAUP* has the additional option to treat gaps like an extra character state (`GAPMODE=NEWSTATE`). These different options can yield different treelengths.
7. If ancestral states (`ANCSTATES`) are defined in PAUP*, they effectively add an ancestral taxon to the tree and may reorient the direction of irreversible or Dollo characters. MacClade does not use `ANCSTATES`.
8. Recall that MacClade and PAUP* both add to the treelength implicit changes within polymorphic terminal taxa. When a terminal taxon is polymorphic with three or more states in a character of user-defined type, MacClade and PAUP* cannot accurately estimate the number of steps required within this terminal taxon for each state that might be ancestral to it, but MacClade does make an attempt to judge at least part of the cost, by adding the smallest change to one of the other states observed for the taxon (see [Chapter 4](#)). MacClade does this in part to prevent assigning impossible states to a terminal node. PAUP* does not do this, and so reports a smaller treelength.
9. MacClade allows the user to mix taxa with polymorphisms (e.g., "0&2") and taxa with uncertain state (e.g., "0/2") together in the same data matrix; by default, PAUP* will interpret all multiple-state listings as one or the other, and thus you will need to change PAUP*'s settings to yield similar results.
10. PAUP* does not have the option to treat polytomies as soft polytomies, that is, as uncertain resolution. If MacClade is considering polytomies to be soft (which is MacClade's default mode), then PAUP* (which always considers polytomies as hard, as if they were multiple speciation) will sometimes yield a higher treelength. If the polytomies arose because PAUP* collapsed zero-length branches, there will be no problem if the characters and assumptions being analyzed in MacClade are the same as those used in the PAUP* analysis. Problems will arise, however, with consensus trees, or if the characters and assumptions being analyzed are different from those used in the PAUP* analysis.
11. If the length of a rooted tree was measured by MacClade for irreversible (or otherwise directed) characters, then the rooted tree is read into PAUP*, and characters are then assigned nondirected transformation types (e.g., unordered), PAUP* will ask to deroot the tree. If it does, and you later reroot the tree for use with directed characters again, PAUP* may place the root in a different place from where it was on the originally read rooted tree. Thus it may get a different treelength on directed characters than MacClade had obtained originally. These problems can be avoided, of course, if you are careful to reread the trees after the directed characters are once again in use.
12. PAUP* automatically adds taxa to fill any tree to include all taxa in the data matrix (except for those explicitly deleted within PAUP*). MacClade allows different trees in the file to contain different subsets of the taxa, and displays them without alteration.

13. MacClade always checks all character states in a step matrix, even those not observed in any taxa, in doing step-matrix calculations. PAUP* can do this, but the default option uses an alternative time-saving measure. Namely, it examines only states in the matrix and any shortcuts identified as such by the 3+1 test. These time-saving measures can in certain circumstances give only approximate treelengths, and therefore PAUP*'s treelength will differ from MacClade's. This may be a common problem if the step matrix violates the triangle inequality, but uncommon otherwise.
14. If MacClade saves a tree with observed taxa fixed as ancestors, it places these taxa in the tree description as named ancestral nodes. PAUP* does not support trees with observed taxa fixed as ancestors. If MacClade saves such a tree, PAUP* will pop out the observed ancestors to terminal position, and therefore may yield a different treelength.

Character reconstructions

The reconstructions of character state evolution shown by MacClade's Trace Character and PAUP*'s Show Reconstructions may differ for the following reasons:

1. Discrepancies described under "Treelength" can also be reflected in differences in character reconstructions between PAUP* and MacClade.
2. The character-changes option of PAUP*'s Show Reconstruction implicitly uses one of the ambiguity-resolving methods, ACCTRAN, DELTRAN, or MINF; in contrast, the possible-state-assignments option does not resolve ambiguity. MacClade's **Trace Character** does not resolve ambiguity unless explicitly requested in **Resolving Options**. Even if both MacClade and PAUP* are using the same resolving method, they may still differ in their character reconstructions because of PAUP*'s choice of only one of the ACCTRAN or DELTRAN reconstructions (see [Chapter 4](#)) and PAUP*'s derooting of trees (see page 475).
3. ACCTRAN/DELTRAN in MacClade apply *only* to the traced character, not to **Trace All Changes** or **Trace All States**; in PAUP* the ambiguity-resolving method is used for branch length, change lists, and so on.
4. PAUP* provides no facility for fixing character states at nodes. If states are fixed at nodes in the character tracing in MacClade, the fixing may not be reflected in the reconstructions shown by PAUP*.

Consistency and retention indices

Users moving between PAUP* and MacClade may notice situations in which the two programs give different consistency and retention indices. Some possible causes are as follows:

1. The treelength or number of character steps may differ (see the two previous sections).
2. MacClade's ensemble consistency indices include only unordered, ordered, and irreversible characters; they do not include stratigraphic, Dollo, and user-defined type characters. Its ensemble retention indices include only unordered and ordered characters; they do not include irreversible, stratigraphic, Dollo and user-defined type characters. PAUP*'s ensemble indices include unordered, ordered, irreversible, and Dollo characters.
3. PAUP versions 3.0a – 3.0g calculated the minimum conceivable number of steps differently from MacClade. PAUP asked, "What is the smallest number of steps that could be obtained with the observed character states?" and thus it effectively allowed polymorphic terminal taxa to be split up into their component states. This approach yields a lower smallest number of steps, and thus lower consistency index than MacClade, which takes the terminal taxa as indivisible and asks:

"What is the smallest number of steps that can be obtained for the character by rearranging the terminal taxa?" MacClade maintains the terminal taxa intact partly because the user's choice to maintain a polymorphic terminal taxon intact implies that the monophyly of the taxon is not to be questioned, and partly because a directly analogous maximum conceivable number of steps can be easily formulated. As with the minimum, MacClade asks for the maximum: "What is the largest number of steps that can be obtained for the character by rearranging the terminal taxa?" PAUP did not ask for the maximum a question analogous to its question for the minimum because the largest number of steps that could be obtained for the observed states in the observed taxa is not a well-defined question when terminal taxa are allowed to break up. Instead, PAUP maintained terminal taxa intact for the maximum, and broke them up for the minimum. A consistency index based on PAUP's minimum is a more complete measure of homoplasy, whereas MacClade's measure can indicate better how consistent characters are to each other. As a result of this distinction, PAUP versions 3.0h and subsequent calculate a "homoplasy index" equal to 1 minus the "consistency index" of earlier versions. The consistency index is now calculated in the same way as does MacClade.

Differences in data file format

Though the NEXUS format was designed to make MacClade and PAUP* data-file compatible, there are some parts of data files that will be accepted by PAUP* but will either not be accepted or not be fully read by MacClade, and vice versa.

Several features in PAUP* NEXUS files are not allowed or are ignored in MacClade:

1. MacClade does not allow abbreviation of tokens.
2. MacClade will not accept all-digit character or taxon names. PAUP* allows them, but warns that taxon/character names have precedence over numbers (e.g., If the tenth character is named "5", MacClade will not read the file; PAUP* will, but the command "EXSET *no_5 = 5;" causes character number 10 to be excluded by PAUP*.) Do not use all-digit names in PAUP* if you intend to use the data file in MacClade (and it's not advisable in PAUP*, in any case).
3. MacClade does not pay attention to uppercase and lowercase (except in a minor way in nucleotide data); thus, RESPECTCASE will be ignored.
4. MacClade will abort reading a file if you use LABELPOS=RIGHT.
5. MacClade will ignore GAPMODE=NEWSTATE.
6. If you use types Dollo.dn or irrev.dn in MacClade, it will convert them to Dollo.up and irrev.up.
7. MacClade does not process the PAUP block.
8. MacClade does not allow you to add extra character state symbols to DNA and RNA data files.
9. PAUP* allows any number of DATA/TAXA/CHARACTERS BLOCKS within a file to allow batch processing of multiple data sets. MacClade allows at most one DATA block, or one TAXA and one CHARACTERS block, and ignores all others, although it will accept two CHARACTERS blocks if one is of datatype CONTINUOUS.

Several features in MacClade NEXUS files are not allowed or are ignored in PAUP:

1. Versions 3.0s and earlier of PAUP will not accept the stratigraphic transformation type ("strat"). Later versions will automatically convert any characters of this type to the default character type, which is unordered unless overridden by a DEFTYPE option setting.
2. PAUP* does not process the CONTINUOUS, CODONS, NOTES, or MACCLADE blocks.
3. As PAUP* treats polytomies only as hard polytomies, PolyTcount=MINSTEPS will be ignored.
4. PAUP* does not allow trees in a trees block to have varying numbers of taxa; to resolve the problem it adds taxa to the trees until each includes all nondeleted taxa.
5. PAUP* does not allow UTREEs and TREEs in the same TREES block.
6. Versions 3.0s and earlier of PAUP will not process a data file with TAXA and CHARACTERS blocks.

There may be other differences we hadn't expected; these should be considered bugs.



REFERENCES

- Archie, J. 1985. Methods for coding variable morphological features for numerical taxonomic analysis. *Syst. Zool.*, 34:326–345.
- Archie, J. 1989a. A randomization test for phylogenetic information in systematic data. *Syst. Zool.*, 38:239–252.
- Archie, J. 1989b. Homoplasy excess ratios: New indices for measuring levels of homoplasy in phylogenetic systematics and a critique of the consistency index. *Syst. Zool.*, 38:239–252.
- Avise, J.C., J.F. Shapiro, S.W. Daniel, C.F. Aquadro, and R.A. Lansman. 1983. Mitochondrial DNA differentiation during the speciation process in *Peromyscus*. *Mol. Biol. Evol.*, 1: 38–56.
- Ball, G. E., and D. R. Maddison. 1987. Classification and evolutionary aspects of the species of the New World genus *Amblygnathus* Dejean, with description of *Platymetopsis*, new genus, and notes about selected species of *Selenophorus* Dejean (Coleoptera: Carabidae: Harpalini). *Trans. Amer. Entomol. Soc.*, 113:189–307.
- Barry, D., and J. A. Hartigan. 1987. Statistical analysis of hominoid molecular evolution. *Stat. Sci.*, 2:191–210.
- Baum, B. R. 1988. A simple procedure for establishing discrete characters from measurement data, applicable to cladistics. *Taxon*, 37:63–70.
- Baum, D.A., and K.L. Shaw. 1995. Genealogical perspectives on the species problem. Pages 289–303 in Experimental and molecular approaches to plant biosystematics (P. C. Hoch and A. G. Stevenson, eds.). Monographs in Systematics, Vol. 53. Missouri Botanical Garden, St. Louis.
- Bremer, K. 1988. The limits of amino acid sequence data in angiosperm phylogenetic reconstruction. *Evolution*, 42:795–803.
- Brooks, D. R., and D. A. McLennan. 1991. *Phylogeny, Ecology and Behavior: A Research Program in Comparative Biology*. Univ. Chicago Press, Chicago.
- Brooks, D. R., R. T. O'Grady, and E. O. Wiley. 1986. A measure of the information content of phylogenetic trees and its use as an optimality criterion. *Syst. Zool.*, 35:571–582.
- Bulmer, M. 1991. Use of generalized least squares in reconstructing phylogenies from sequence data. *Mol. Biol. Evol.*, 8:868–883.
- Camin, J. H., and R. R. Sokal. 1965. A method for deducing branching sequences in phylogeny. *Evolution*, 19:311–326.
- Cann, R. L., M. Stoneking, and A. C. Wilson. 1987. Mitochondrial DNA and human evolution. *Nature*, 325:31–36.
- Carpenter, J. M. 1988. Choosing among multiple equally parsimonious cladograms. *Cladistics*, 4:291–296.
- Cavalli-Sforza, L. L., and A. W. F. Edwards. 1967. Phylogenetic analysis: Models and estimation procedures. *Evolution*, 32:550–570.
- Cavender, J. A. 1978. Taxonomy with confidence. *Math. Biosci.*, 40:271–280.
- Cavender, J. A. 1981. Tests of phylogenetic hypotheses under generalized models. *Math. Biosci.*, 54:217–229.
- Cavender, J. A. 1989. Mechanized derivation of linear invariants. *Mol. Biol. Evol.*, 6:301–316.

- Cavender, J. A., and J. Felsenstein. 1987. Invariants of phylogenies in a simple case with discrete states. *J. Class.*, 4:57–71.
- Chang, B. S. W. and M. J. Donoghue. 2000. Recreating ancestral proteins. *Trends Ecol. Evol.* 15:109–114
- Chappill, J. A. 1989. Quantitative characters in phylogenetic analysis. *Cladistics*, 5:217–234.
- Coddington, J. A. 1988. Cladistic tests of adaptational hypotheses. *Cladistics*, 4:3–22.
- Cunningham, C. W., K. E. Omland, and T. H. Oakley. 1998. Reconstructing ancestral character states: A critical reappraisal. *Trends Ecol. evol.* 13:361–366.
- DeBry, R. W., and N. A. Slade. 1985. Cladistic analysis of restriction endonuclease cleavage maps within a maximum-likelihood framework. *Syst. Zool.*, 34:21–34.
- de Queiroz, K. 1996. Including the characters of interest during tree reconstruction and the problems of circularity and bias in studies of character evolution. *Am. Nat.*, 148: 700–708
- de Queiroz, K., and M. J. Donoghue. 1988. Phylogenetic systematics and the species problem. *Cladistics*, 4:317–338.
- de Queiroz, K., and M. J. Donoghue. 1990. Phylogenetic systematics or Nelson's version of cladistics? *Cladistics*, 6:61–75.
- Donoghue, M. J. 1989. Phylogenies and the analysis of evolutionary sequences, with examples from seed plants. *Evolution*, 43:1137–1156.
- Donoghue, M. J., and J. A. Doyle, 1989. Phylogenetic analysis of angiosperms and the relationships of Hamamelidae. Pp. 17–45 in P. R. Crane and S. Blackmore (eds.), *Evolution, Systematics and Fossil History of the Hamamelidae, Volume 1: Introduction and 'Lower Hamamelidae'*. Systematics Assoc., Special Vol. No. 40A, London.
- Donoghue, M. J., and M. J. Sanderson. 1992. The suitability of molecular and morphological evidence in reconstructing plant phylogeny. Pp. 340–368 in P. S. Soltis, D. E. Soltis, and J. J. Doyle (eds.), *Molecular Systematics of Plants*. Chapman and Hall, N.Y.
- Donoghue, M. J., R. G. Olmstead, J. F. Smith, and J. D. Palmer. 1992. Phylogenetic relationships of the Dipsacales based on rbcL sequences. *Ann. Missouri Bot. Gard.*, 79:333–345.
- Doolittle, W. F. 1999. Phylogenetic classification and the universal tree. *Science*, 284:2124–2128.
- Doyle, J. A., and M. J. Donoghue. 1986. Seed plant phylogeny and the origin of angiosperms: An experimental cladistic approach. *Bot. Rev.*, 52:321–431.
- Doyle, J. A., S. Jardiné, and A. Doerenkamp. 1982. *Afropollis*, a new genus of early angiosperm pollen, with notes on the Cretaceous palynostratigraphy and paleoenvironments of Northern Gondwana. *Bull. Centres Rech. Expl. Prod. Elf-Aquitaine*, 6:39–117.
- Doyle, J.J. 1992. Gene trees and species trees: Molecular systematics as one-character taxonomy. *Syst. Bot.* 17: 144–163.
- Edwards, A. W. F. 1970. Estimating the branch points of a branching diffusion process. *J. Roy. Statist. Soc.*, B32:154–174.
- Edwards, A. W. F., and L. L. Cavalli-Sforza. 1964. Reconstruction of evolutionary trees. Pp. 67–76 in V. H. Heywood and J. McNeill (eds.), *Phenetic and Phylogenetic Classification*. Systematics Assoc., Publ. No. 6, London.
- Eldredge, N., and J. Cracraft. 1980. *Phylogenetic Patterns and the Evolutionary Process: Method and Theory in Comparative Biology*. Columbia Univ. Press, N.Y.
- Elzanowski, A., and J. Ostell. 2000. The Genetic Codes. <http://www.ncbi.nlm.nih.gov/htbin-post/Taxonomy/wprintgc?mode=c>.
- Estabrook, G. F., C. S. Johnson Jr., and F. R. McMorris. 1976. A mathematical foundation for the analysis of cladistic character compatibility. *Math. Biosci.* 29:181–187.

- Farris, J. S. 1969. A successive approximations approach to character weighting. *Syst. Zool.*, 18:374–385.
- Farris, J. S. 1970. Methods for computing Wagner Trees. *Syst. Zool.*, 19:83–92.
- Farris, J. S. 1972. Estimating phylogenetic trees from distance matrices. *Am. Nat.*, 106:645–668.
- Farris, J. S. 1977. Phylogenetic analysis under Dollo's Law. *Syst. Zool.*, 26:77–88.
- Farris, J. S. 1978. Inferring phylogenetic trees from chromosome inversion data. *Syst. Zool.*, 27:275–284.
- Farris, J. S. 1982. Outgroups and parsimony. *Syst. Zool.*, 31:328–334.
- Farris, J. S. 1983. The logical basis of phylogenetic analysis. Pp. 7–36 in N. I. Platnick and V. A. Funk (eds.), *Advances in Cladistics*, Vol. 2. Proceedings of the Second Meeting of the Willi Hennig Society. Columbia Univ. Press, N.Y.
- Farris, J. S. 1988. *HENNIG86, version 1.5*. Distributed by the author, Port Jefferson Station, N.Y.
- Farris, J. S. 1989. The retention index and the rescaled consistency index. *Cladistics*, 5:417–419.
- Felsenstein, J. 1973. Maximum-likelihood estimation of evolutionary trees from continuous characters. *Am. J. Hum. Genet.*, 25:471–492.
- Felsenstein, J. 1978a. The number of evolutionary trees. *Syst. Zool.*, 27:27–33.
- Felsenstein, J. 1978b. Cases in which parsimony or compatibility methods will be positively misleading. *Syst. Zool.*, 27:401–410.
- Felsenstein, J. 1979. Alternative methods of phylogenetic inference and their interrelationship. *Syst. Zool.*, 28:49–62.
- Felsenstein, J. 1981a. A likelihood approach to character weighting and what it tells us about parsimony and compatibility. *Biol. J. Linn. Soc.*, 16:183–196.
- Felsenstein, J. 1981b. Evolutionary trees from DNA sequences: A maximum likelihood approach. *J. Mol. Evol.*, 17:368–376.
- Felsenstein, J. 1982. Numerical methods for inferring evolutionary trees. *Q. Rev. Biol.*, 57:379–404.
- Felsenstein, J. 1985a. Phylogenies and the comparative method. *Am. Nat.*, 125:1–15.
- Felsenstein, J. 1985b. Confidence limits on phylogenies with a molecular clock. *Syst. Zool.*, 34:152–161.
- Felsenstein, J. 1985c. Confidence limits on phylogenies: An approach using the bootstrap. *Evolution*, 39:783–791.
- Felsenstein, J. 1988a. Phylogenies from molecular sequences: Inference and reliability. *Ann. Rev. Genet.*, 22:521–565.
- Felsenstein, J. 1988b. Phylogenies and quantitative characters. *Ann. Rev. Ecol. Syst.*, 19:445–471.
- Felsenstein, J. 1992. Phylogenies from restriction sites: A maximum-likelihood approach. *Evolution*, 46:159–173.
- Felsenstein, J. 1993. *PHYLIP (Phylogeny Inference Package) version 3.5c*. Distributed by the author. Department of Genetics, University of Washington, Seattle.
- Fisher, D. C. 1982. Phylogenetic and macroevolutionary patterns within the Xiphosurida. *N. Amer. Paleont. Conv. III, Proc.*, 1:175–180.
- Fisher, D. C. 1988. Stratocladistics: Integrating stratigraphic and morphologic data in phylogenetic inference. *Geol. Soc. Amer. Abst. Prog.*, 20:A186.
- Fisher, D. C. 1991. Phylogenetic analysis and its application in evolutionary paleobiology. Pp. 103–122 in N. L. Gilinsky and P. W. Signor (eds.), *Analytical Paleobiology*. Short Courses in Paleontology, No. 4, Paleontological Society.
- Fisher, D. C. 1992. Stratigraphic parsimony. Pp. 124–129 in W.P. Maddison and D.R. Maddison, *MacClade: Analysis of Phylogeny and Character Evolution*. Sinauer Associates, Sunderland, Mass.

- Fishman, G. S., and L. R. Moore. 1982. A statistical evaluation of multiplicative congruential random number generators with modulus $2^{31} - 1$. *J. Amer. Stat. Assoc.*, 77:129–136.
- Fitch, W. M. 1971. Toward defining the course of evolution: Minimal change for a specific tree topology. *Syst. Zool.*, 20:406–416.
- Fitch, W. M. 1979. Cautionary remarks on using gene expression events in parsimony procedures. *Syst. Zool.*, 28:375–379.
- Fitch, W. M. 1986. A hidden bias in the estimate of total nucleotide substitutions from pairwise differences. Pp. 315–328 in S. Karlin and E. Nevo (eds.), *Evolutionary Process and Theory*. Academic Press, Orlando, Florida.
- Fitch, W. M., and J. J. Bientema. 1990. Correcting parsimonious trees for unseen nucleotide substitutions: The effect of dense branching exemplified by ribonuclease. *Mol. Biol. Evol.*, 7:438–443.
- Fitch, W. M., and M. Bruschi. 1987. The evolution of prokaryotic ferredoxins — with a general method correcting for unobserved substitutions in less branches lineages. *Mol. Biol. Evol.*, 4:381–394.
- Fitch, W. M., and J. S. Farris. 1974. Evolutionary trees with minimum nucleotide replacements from amino acid sequences. *J. Mol. Evol.*, 3:263–278.
- Fitch, W. M., and E. Margoliash. 1967. Construction of phylogenetic trees. *Science*, 155:279–284.
- Fitch, W. M., and J. Ye. 1992. Weighted parsimony: Does it work? Pp. 147–154 in M. M. Miyamoto and J. Cracraft (eds.), *Phylogenetic Analysis of DNA Sequences*. Oxford Univ. Press, Oxford.
- Frumhoff, P. C. and H. K. Reeve. 1994. Using phylogenies to test hypotheses of adaptation: A critique of some current proposals. *Evolution* 48:172–180.
- Gojobori, T., W.-H. Li, and D. Graur. 1982. Patterns of nucleotide substitution in pseudogenes and functional genes. *J. Mol. Evol.*, 18:360–369.
- Golding, B., and J. Felsenstein. 1990. A maximum likelihood approach to detection of selection from a phylogeny. *J. Mol. Evol.*, 31:511–523.
- Goldman, N. 1988. Methods for discrete coding of morphological characters in numerical analysis. *Cladistics*, 4:59–71.
- Goldman, N. 1990. Maximum likelihood inference of phylogenetic trees, with special reference to a Poisson process model of DNA substitution and to parsimony analysis. *Syst. Zool.*, 39:345–361.
- Goldman, N. 1993a. Simple diagnostic statistical tests of models of DNA substitutions. *J. Mol. Evol.* 37:650–661.
- Goldman, N. 1993b. Statistical tests of models of DNA evolution. *J. Mol. Evol.* 36:182–198.
- Goldman, N., and S. Whelan. 2000. Statistical tests of gamma-distributed rate heterogeneity in models of sequence evolution in phylogenetics. *Mol. Biol. Evol.* 17:975–978
- Goldman, N., and Z. H. Yang. 1994. Codon-based model of nucleotide substitution for protein-coding DNA sequences. *Mol. Biol. Evol.* 11:725–736
- Goloboff, P.A. 1999. *NONA, version 2.0*. Distributed by the author, Instituto Miguel Lillo, Tucuman, Argentina.
- Goodman, M. 1981. Globin evolution was apparently very rapid in early vertebrates: A reasonable case against the rate-constancy hypothesis. *J. Mol. Evol.*, 17:114–120.
- Gotoh, O. 1982. An improved algorithm for matching biological sequences. *J. Mol. Biol.*, 162:705–708.
- Griswold, C. E. 1987. A revision of the jumping spider genus *Habronattus* F. O. P.-Cambridge (Araneae; Salticidae), with phenetic and cladistic analyses. *Univ. Calif. Publ. Entomol.*, 107:1–344.
- Harding, E. F. 1971. The probabilities of rooted tree-shapes generated by random bifurcation. *Adv. Appl. Prob.*, 3:44–77.

- Hartigan, J. A. 1973. Minimum mutation fits to a given tree. *Biometrics*, 29:53–65.
- Harvey, P. H., A. J. Leigh Brown and J. Maynard Smith. 1995. New uses for phylogenies: editor's introduction. *Phil. Trans. R. Soc. London B* 349:3–4.
- Harvey, P. H., and M. D. Pagel. 1991. *The Comparative Method in Evolutionary Biology*. Oxford Univ. Press, Oxford.
- Hayasaka, K., T. Gojobori, and S. Horai. 1988. Molecular phylogeny and evolution of primate mitochondrial DNA. *Mol. Biol. Evol.*, 5:626–644.
- Hein, J. 1989. A new method that simultaneously aligns and reconstructs ancestral sequences for any number of homologous sequence, when the phylogeny is given. *Mol. Biol. Evol.*, 6:649–668.
- Hein, J. 1990. TreeAlign. Source code for a program, distributed by the author.
- Hendy, M.D. 1991. A combinatorial description of the closest tree algorithm for finding evolutionary trees. *Discrete Mathematics*, 96:51–18.
- Hendy, M. D., and D. Penny. 1982. Branch and bound algorithms to determine minimal evolutionary trees. *Math. Biosci.*, 59:277–290.
- Hendy, M. D., and D. Penny. 1984. Cladograms should be called trees. *Syst. Zool.*, 33:245–247.
- Hendy, M. D., and D. Penny. 1989. A framework for the quantitative study of evolutionary trees. *Syst. Zool.*, 38:297–309.
- Hennig, W. 1966. *Phylogenetic Systematics*. Univ. Illinois Press, Urbana.
- Hillis, D. M. 1987. Molecular versus morphological approaches to systematics. *Ann. Rev. Ecol. Syst.*, 18:23–42.
- Hillis, D. M., and M. T. Dixon. 1989. Vertebrate phylogeny: Evidence from 28S ribosomal DNA sequences. Pp. 355–367 in B. Fernholm, K. Bremer, and H. Jörnvall (eds.), *The Hierarchy of Life*. Elsevier Press, Amsterdam.
- Hillis, D. M., J. J. Bull, M. E. White, M. R. Badgett, and I. J. Molineux. 1992. Experimental phylogenetics: Generation of a known phylogeny. *Science*, 255:589–592.
- Hillis, D.M., J.P. Huelsenbeck, and C.W. Cunningham. 1994. Application and accuracy of molecular phylogenies. *Science*, 264:671–677.
- Holmquist, R. 1979. The method of parsimony: An experimental test and theoretical analysis of the adequacy of molecular restoration studies. *J. Mol. Biol.*, 135:939–958.
- Huelsenbeck, J. P. 1991. Tree-length distribution skewness: An indicator of phylogenetic information. *Syst. Zool.*, 40:257–270.
- Huelsenbeck, J.P., and K. A. Crandall. 1997. Phylogeny estimation and hypothesis testing using maximum likelihood. *Ann. Rev. Ecol. Syst.* 28:437–466
- Huelsenbeck, J. P., and B. Rannala. 1997. Phylogenetic methods come of age: Testing hypotheses in an evolutionary context. *Science* 276(5310):227–232
- Huelsenbeck J. P., B. Larget, and D. Swofford. 2000. A compound Poisson process for relaxing the molecular clock. *Genetics* 154:1879–1892.
- Huey, R. B., and A. F. Bennett. 1987. Phylogenetic studies of coadaptation: Preferred temperatures versus optimal performance temperatures of lizards. *Evolution*, 41:1098–1115.
- Hull, D. L. 1967. Certainty and circularity in evolutionary taxonomy. *Evolution*, 21:174–189.
- Hull, D. L. 1979. The limits of cladism. *Syst. Zool.*, 28:416–440.

- Iwabe, M., K. Kuma, M. Hasegawa, S. Osawa, and T. Miyata. 1989. Evolutionary relationship of archaebacteria, eubacteria and eukaryotes inferred from phylogenetic tree of duplicated genes. *Proc. Natl. Acad. Sci. USA*, 86:9355–9359.
- Janson, C. H. 1992. Measuring evolutionary constraints: A markov model for phylogenetic transitions among seed dispersal syndromes. *Evolution*, 46:136–158.
- Jin, L., and M. Nei. 1990. Limitations of the evolutionary parsimony method of phylogenetic analysis. *Mol. Biol. Evol.*, 7:82–102.
- Kimura, M. 1981. Doubt about studies of globin evolution based on maximum parsimony codons and the augmentation procedure. *J. Mol. Evol.*, 17:121–122.
- Kishino, H., T. Miyata, and M. Hasegawa. 1990. Maximum likelihood inference of protein phylogeny and the origin of chloroplasts. *J. Mol. Evol.*, 31:151–160.
- Kluge, A. G., and J. S. Farris. 1969. Quantitative phyletics and the evolution of the anurans. *Syst. Zool.*, 18:1–32.
- Koshi, J. M., and R. A. Goldstein. 1996. Probabilistic reconstruction of ancestral protein sequences. *J. Mol. Evol.*, 42:313–320.
- Kumar, S., K. Tamura, and M. Nei. 1993. *MEGA: Molecular Evolutionary Genetics Analysis*, version 1.0. The Pennsylvania State University, University Park, PA 16802.
- Lake, J. A. 1987. A rate-independent technique for analysis of nucleic acid sequences: Evolutionary parsimony. *Mol. Biol. Evol.*, 4:167–191.
- Lake, J. A. 1994. Reconstructing evolutionary trees from DNA and protein sequences: Paralinear distances. *Proc. Natl. Acad. Sci. USA*, 91:1455–1459.
- Lanave, C., G. Preparata, C. Saccone, and G. Serio. 1984. A new method for calculating evolutionary substitution rates. *J. Mol. Evol.*, 20:86–93.
- Le Quesne, W. J. 1974. The uniquely evolved character concept and its cladistic application. *Syst. Zool.*, 23:513–517.
- Lee, M.S.Y. 1999. Circularity, evolution, systematics ... and circularity. *J. Evol. Biol.*, 12: 724–734.
- Liebherr, J. K., and A. E. Hajek. 1990. A cladistic test of the taxon cycle and taxon pulse hypotheses. *Cladistics*, 6:39–59.
- Lio, P., and N. Goldman. 1998. Models of molecular evolution and phylogeny. *Genome Research*, 12:1233–1244.
- Lockhart, P.J., M.A. Steel, M.D. Hendy, and D. Penny. 1994. Recovering evolutionary trees under a more realistic model of sequence evolution. *Mol. Biol. Evol.*, 11:605–612.
- Lorch, P. D., and J. M. Eadie. 1999. Power of the concentrated changes test for correlated evolution. *Syst. Biol.* 48:170–191.
- Losos, J. 1995. An approach to the analysis of comparative data when a phylogeny is unavailable or incomplete. *Syst. Biol.* 43:117–123.
- Maddison, D. R. 1990. Phylogenetic inference of historical pathways and models of evolutionary change. PhD. thesis, Harvard Univ.
- Maddison, D. R. 1991a. The discovery and importance of multiple islands of most-parsimonious trees. *Syst. Zool.*, 40:315–328.
- Maddison, D. R. 1991b. African origin of human mitochondrial DNA reexamined. *Syst. Zool.*, 40:355–363.
- Maddison, D. R. 1993. Systematics of the Holarctic ground-beetle subgenus *Bracteon* Netolitzky and related *Bembidion* Latreille (Coleoptera: Carabidae). *Bull. Mus. Comp. Zool.*, 153:143–299.

- Maddison, D.R. 1994. Phylogenetic methods for inferring the evolutionary history and processes of change in discretely valued characters. *Annu. Rev. Entomol.*, 39:267–292.
- Maddison, D. R., M. Ruvolo, and D. L. Swofford. 1992. Geographic origins of human mitochondrial DNA: Phylogenetic evidence from control region sequences. *Syst. Biol.*, 41:111–124.
- Maddison, D.R., D.L. Swofford, and W.P. Maddison. 1997. NEXUS: An extensible file format for systematic information. *Systematic Biology*, 46:590–621.
- Maddison, W. P. 1982. XXXY sex chromosomes in males of the jumping spider genus *Pellenes* (Araneae: Salticidae). *Chromosoma (Berlin)*, 85:23–37.
- Maddison, W. P. 1986. *MacClade 1.0*. Distributed by the author, Cambridge, Mass.
- Maddison, W. P. 1989. Reconstructing character evolution on polytomous cladograms. *Cladistics*, 5:365–377.
- Maddison, W. P. 1990. A method for testing the correlated evolution of two binary characters: Are gains or losses concentrated on certain branches of a phylogenetic tree? *Evolution*, 44:539–557.
- Maddison, W. P. 1991. Squared-change parsimony reconstructions of ancestral states for continuous-valued characters on a phylogenetic tree. *Syst. Zool.*, 40:304–314.
- Maddison, W.P. 1993. Missing data versus missing characters in phylogenetic analysis. *Systematic Biology*. 42: 576–581.
- Maddison, W.P. 1995. Calculating the probability distributions of ancestral states reconstructed by parsimony on phylogenetic trees. *Syst. Biol.*, 44:474–481.
- Maddison, W.P. 1996. Molecular approaches and the growth of phylogenetic biology. Pages 47–63 in J.D. Ferraris and S.R. Palumbi (eds.). *Molecular zoology: Advances, strategies, and protocols*. Wiley-Liss, New York.
- Maddison, W.P. 1997. Gene trees in species trees. *Syst. Biol.*, 46:523–536.
- Maddison, W.P. 2000. Testing character correlation using pairwise comparisons on a phylogeny. *J. Theoretical Biology*. 202: 195–204.
- Maddison, W. P., and D. R. Maddison. 1987. *MacClade 2.1*. Distributed by the authors, Cambridge, Mass.
- Maddison, W. P., and D. R. Maddison. 1992. *MacClade version 3: Analysis of phylogeny and character evolution*. Sinauer Associates, Sunderland Massachusetts.
- Maddison, W. P., and M. Slatkin. 1990. Parsimony reconstructions of ancestral states do not depend on the relative distances between linearly-ordered character states. *Syst. Zool.*, 39:175–178.
- Maddison, W. P., and M. Slatkin. 1991. Null models for the number of evolutionary steps in a character on a phylogenetic tree. *Evolution*, 45:1184–1197.
- Maddison, W. P., M. J. Donoghue, and D. R. Maddison. 1984. Outgroup analysis and parsimony. *Syst. Zool.*, 33:83–103.
- Martins, E.P. 1996. Conducting phylogenetic comparative studies when the phylogeny is not known. *Evolution* 50:12–22.
- Martins, E. P. 1999. Estimation of ancestral states of continuous characters: A computer simulation study. *Syst. Biol.* 48:642–650.
- Martins, E., and T. Garland. 1991. Phylogenetic analyses of the evolution of continuous characters: A simulation study. *Evolution*, 45:534–557.
- Mayr, E. 1963. *Animal Species and Evolution*. Harvard Univ. Press, Cambridge, Mass.
- McArdle, B. and A. G. Rodrigo. 1994. Estimating the ancestral states of a continuous-valued character using squared-change parsimony: An analytical solution. *Syst. Biol.* 43:573–578.

- Meacham, C. A. 1984. The role of hypothesized direction of characters in the estimation of evolutionary history. *Taxon*, 33:26–38.
- Meacham, C. A., and G. Estabrook. 1985. Compatibility methods in systematics. *Ann. Rev. Ecol. Syst.*, 16:431–446.
- Mickevich, M. F. 1981. Quantitative phylogenetic biogeography. Pp. 209–222 in V. A. Funk and D. R. Brooks (eds.), *Advances in Cladistics*. Proceedings of the First Meeting of the Willi Hennig Society. The New York Botanical Garden, N.Y.
- Mickevich, M. F. 1982. Transformation series analysis. *Syst. Zool.*, 31:461–478.
- Mickevich, M. F., and M. S. Johnson. 1976. Congruence between morphological and allozyme data in evolutionary inference and character evolution. *Syst. Zool.*, 25:260–270.
- Mickevich, M. F., and C. Mitter. 1981. Treating polymorphic character in systematics: A phylogenetic treatment of electrophoretic data. Pp. 45–58 in V. A. Funk and D. R. Brooks (eds.), *Advances in Cladistics*. Proceedings of the First Meeting of the Willi Hennig Society. The New York Botanical Garden, N.Y.
- Mickevich, M. F., and S. J. Weller. 1990. Evolutionary character analysis: Tracing character change on a cladogram. *Cladistics*, 6:137–170.
- Mitter, C. 1981. "Cladistics" in botany. *Syst. Zool.*, 30:373–376.
- Miyamoto, M. M., and S. M. Boyle. 1989. The potential importance of mitochondrial DNA sequence data to eutherian mammal phylogeny. Pp. 437–450 in B. Fernholm, K. Bremer, and H. Jörnvall (eds.), *The Hierarchy of Life*. Elsevier Press, Amsterdam.
- Mooers, A.Ø. and D. Schlüter. 1999. Reconstructing ancestral states with maximum likelihood: support for one- and two-rate models. *Syst. Biol.*, 48:623–633.
- Mooi, R. 1989. The outgroup criterion revisited via naked zones and alleles. *Syst. Zool.*, 38:283–290.
- Moore, G. W., Barnabas, J. and M. Goodman. 1973. A method for constructing maximum parsimony ancestral amino acid sequences on a given network. *J. Theor. Biol.*, 38:459–485.
- Navidi, W. C., G. A. Churchill, and A. von Haeseler. 1991. Methods for inferring phylogenies from nucleic acid sequence data by using maximum likelihood and linear invariants. *Mol. Biol. Evol.*, 8:128–143.
- Needleman, S.B., and Wunsch, C.D. 1970. A general method applicable to the search for similarities in the amino-acid sequence of two proteins. *J. Mol. Biol.*, 48:443–453.
- Neff, N. A. 1986. A rational basis for a priori character weighting. *Syst. Zool.*, 35:110–123.
- Nei, M. 1987. *Molecular Evolutionary Genetics*. Columbia Univ. Press, N.Y.
- Nelson, G. 1974. Classification as an expression of phylogenetic relationships. *Syst. Zool.*, 22:344–359.
- Nelson, G., and N. I. Platnick. 1981. *Systematics and Biogeography: Cladistics and Vicariance*. Columbia Univ. Press, N.Y.
- Nixon, K. C. 1992. *CLADOS, version 1.2*. Distributed by the author, Ithaca, N.Y.
- Nixon, K. C., and J. I. Davis. 1991. Polymorphic taxa, missing values and cladistic analysis. *Cladistics*, 7:233–241.
- Nixon, K. C., and Q. D. Wheeler. 1990. An amplification of the phylogenetic species concept. *Cladistics*, 6:211–223.
- O'Grady, R. T., and G. B. Deets. 1987. Coding multistate characters, with special reference to the use of parasites as characters of their hosts. *Syst. Zool.*, 36:268–279.
- Olsen, G. J. 1991. Systematic underestimation of tree branch lengths by Lake's operator metrics: An effect of position-dependent substitution rates. *Mol. Biol. Evol.*, 8:592–608.
- Omland, K. E. 1999. The assumptions and challenges of ancestral state reconstruction. *Syst. Biol.*, 48:604–611.

- Page, R. D. M. 1993. *COMPONENT version 2: Tree comparison software for Microsoft Windows*. Natural History Museum, London.
- 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. 1997. Inferring evolutionary processes from phylogenies. *Zoological Scripta*. 26: 331–348
- Pagel, M. 1999a. Inferring the historical patterns of biological evolution. *Nature* 401(6756):877–884
- Pagel, M. 1999b. The maximum likelihood approach to reconstructing ancestral states of discrete characters on phylogenies. *Syst. Biol.* 48:612–622.
- Pagel, M. D., and P. H. Harvey. 1989. Comparative methods for examining adaptation depend on evolutionary methods. *Folia Primatologica*, 53:203–220.
- Patterson, C. 1982. Morphological characters and homology. Pp. 21–74 in K. Joysey and A. Friday (eds.), *Problems of Phylogenetic Reconstruction*. Academic Press, London.
- Payne, W. H., J. R. Rabung, and T. P. Bogyo. 1969. Coding the Lehmer pseudo-random number generator. *Comm. ACM*, 12:85–86.
- Penny, D., M.D. Hendy, and M.A. Steel. 1992. Progress with methods for constructing evolutionary trees. *Trends in Ecology and Evolution*, 7:73–79.
- Platnick, N. I. 1977. Cladograms, phylogenetic trees, and hypothesis testing. *Syst. Zool.*, 26:438–442.
- Platnick, N. I. 1986. On justifying cladistics. *Cladistics*, 2:83–85.
- Platnick, N. I., C. E. Griswold, and J. A. Coddington. 1991. On missing entries in cladistic analysis. *Cladistics*, 7:337–343.
- Pogue, M. G., and M. F. Mickevich. 1990. Character definitions and character state delineation: The bête noire of phylogenetic inference. *Cladistics*, 6:319–361.
- Read, A. F., and S. Nee. 1995. Inference from binary comparative data. *J. Theor. Biol.* 173:99–108
- Rempe, U. 1988. Characterizing DNA variability by stochastic matrices. Pp. 375–384 in H.H. Bock (ed.), *Classification and Related Methods of Data Analysis*. Elsevier, North Holland.
- Ridley, M. 1983. *The Explanation of Organic Diversity*. Oxford Univ. Press, Oxford.
- Ridley, M. and A. Grafen. 1996. How to study discrete comparative methods. Pp. 76–103 in E. P. Martins (ed.), *Phylogenies and the comparative method in animal behavior*. Oxford Univ. Press.
- Rinsma, I., Hendy, M., and Penny, D. 1990. Minimally colored trees. *Math. Biosci.*, 98:201–210.
- Ritland, K., and M. T. Clegg. 1987. Evolutionary analysis of plant DNA sequences. *Am. Nat.*, 130:S74–S100.
- Rogers, J. S. 1984. Deriving phylogenetic trees from allele frequencies. *Syst. Zool.*, 33:52–63.
- Rzhetsky, A., and M. Nei. 1992. A simple method for estimating and testing minimum-evolution trees. *Mol. Biol. Evol.*, 9:945–967.
- Saitou, N. 1989. A theoretical study of the underestimation of branch lengths by the maximum parsimony principle. *Syst. Zool.*, 38:1–6.
- Saitou, N., and M. Nei. 1987. The neighbor-joining method: A new method for reconstructing phylogenetic trees. *Mol. Biol. Evol.*, 4:406–425.
- Sanderson, M. J. 1989. Confidence limits on phylogenies: The bootstrap revisited. *Cladistics*, 5:113–129.
- Sanderson, M. J. 1991. In search of homoplastic tendencies: Statistical inference of topological patterns in homoplasy. *Evolution*, 45:351–358.

- Sanderson, M. J. 1993. Reversibility in evolution: A maximum likelihood approach to character gain-loss bias in phylogenies. *Evolution* 47:236–252.
- Sankoff, D. 1990. Designer invariants for large phylogenies. *Mol. Biol. Evol.*, 7:255–269.
- Sankoff, D., and R. J. Cedergren. 1983. Simultaneous comparison of three or more sequences related by a tree. Pp. 253–263 in D. Sankoff and J. B. Kruskal (eds.), *Time Warps, String Edits, and Macromolecules: The Theory and Practice of Sequence Comparison*. Addison-Wesley, Reading, Mass.
- Sankoff, D., and P. Rousseau. 1975. Locating the vertices of a Steiner tree in arbitrary space. *Math. Progr.*, 9:240–246.
- Schlüter, D. 1995. Uncertainty in ancient phylogenies. *Nature* 377:108–109.
- Schlüter, D., T. Price, A. Mooers, and D. Ludwig. 1997. Likelihood of ancestral states in an adaptive radiation. *Evolution* 51:1699–1711.
- Schultz, T. R. and G. A. Churchill. 1999. The role of subjectivity in reconstructing ancestral character states: A Bayesian approach to unknown rates, states and transformation asymmetries. *Syst. Biol.* 48:651–664.
- Schultz, T. R., R. B. Crocroft, and G. A. Churchill. 1996. The reconstruction of ancestral character states. *Evolution* 50:504–511.
- Sidow, A., and A. C. Wilson. 1990. Compositional statistics: An improvement of evolutionary parsimony and its application to deep branches in the tree of life. *J. Mol. Evol.*, 31:51–68.
- Sidow, A., and A. C. Wilson. 1992. Composititional statistics evaluated by computer simulations. Pp. 129–146 in M. M. Miyamoto and J. Cracraft (eds.), *Phylogenetic Analysis of DNA Sequences*. Oxford Univ. Press, Oxford.
- Sillén-Tullberg, B. 1993. The effect of biased inclusion of taxa on the correlation between discrete characters in phylogenetic trees. *Evolution* 47:1182–1191.
- Simberloff, D., K. L. Heck, E. D. McCoy, and E. F. Connor. 1981. There have been no statistical tests of cladistic biogeographical hypotheses. Pp. 40–63 in G. Nelson and D. E. Rosen (eds.), *Vicariance Biogeography: A Critique*. Columbia Univ. Press, N.Y.
- Slatkin, M., and W. P. Maddison. 1989. A cladistic measure of gene flow inferred from the phylogenies of alleles. *Genetics*, 123:603–613.
- Sneath, P. H. A., and R. R. Sokal. 1973. *Numerical Taxonomy: The Principles and Practice of Numerical Classification*. W. H. Freeman, San Francisco.
- Sober, E. 1988. *Reconstructing the Past: Parsimony, Evolution, and Inference*. MIT Press, Cambridge, Mass.
- Sokal, R. R., and P. H. Sneath. 1963. *Principles of Numerical Taxonomy*. W. H. Freeman, San Francisco.
- Steel, M. 1994. Recovering a tree from the Markov leaf colourations it generates under a Markov model. *Appl. Math. Lett.*, 7:19–23.
- Steel, M., and D. Penny. 2000. Parsimony, likelihood, and the role of models in molecular phylogenetics. *Mol. Biol. Evol.* 17:839–850
- Stevens, P. F. 1991. Character states, morphological variation, and phylogenetic analysis: A review. *Syst. Bot.*, 16:553–583.
- Swofford, D. L. 1991. *Phylogenetic Analysis Using Parsimony (PAUP), version 3.0s*. Illinois Natural History Survey, Champaign.
- Swofford, D. L. 2000. *PAUP* 4*. Sinauer Associates, Sunderland, Mass.
- Swofford, D. L., and S. H. Berlocher. 1987. Inferring evolutionary trees from gene frequency data under the principle of maximum parsimony. *Syst. Zool.*, 36:293–325.

- Swofford, D. L., and W. P. Maddison. 1987. Reconstructing ancestral character states under Wagner parsimony. *Math. Biosci.*, 87:199–229.
- Swofford, D. L., and W. P. Maddison. 1992. Parsimony, character-state reconstructions, and evolutionary inferences. Pages 187 – 223. In R. L. Mayden (ed.), *Systematics, Historical Ecology, and North American Freshwater Fishes*. Stanford Univ. Press, Stanford, California.
- Swofford, D. L., and G. J. Olsen. 1990. Phylogeny reconstruction. Pp. 411–501 in D. M. Hillis and G. Moritz (eds.), *Molecular Systematics*. Sinauer Associates, Sunderland, Mass.
- Swofford, D. L., G.J. Olsen, P.J. Waddell, and D.M. Hillis. 1996. Pp. 407–514 in D. M. Hillis, C. Moritz, and B.K. Mable (eds.), *Molecular Systematics*, Second Edition. Sinauer Associates, Sunderland, Mass.
- Tajima, F. 1992. Statistical method for estimating the standard errors of branch lengths in a phylogenetic tree reconstructed without assuming equal rates of substitution among different lineages. *Mol. Biol. Evol.*, 9:168–181.
- Tateno, Y. 1990. A method for molecular phylogeny construction by direct use of nucleotide sequence data. *J. Mol. Evol.*, 30:85–93.
- Templeton, A. R. 1983. Phylogenetic inference from restriction endonuclease cleavage site maps with particular reference to the evolution of humans and the apes. *Evolution*, 37:221–244.
- Thomas, R.H., W. Schaffner, A.C. Wilson, and S. Pääbo. 1989. DNA phylogeny of the extinct marsupial wolf. *Nature*, 340:465–467.
- Thompson, E. A. 1975. *Human Evolutionary Trees*. Cambridge Univ. Press, Cambridge.
- Thompson, E. A. 1986. Likelihood and parsimony: Comparison of criteria and solutions. *Cladistics*, 2:43–52.
- Thompson, J.D., Higgins, D.G. and Gibson, T.J. 1994. CLUSTAL W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, positions-specific gap penalties and weight matrix choice. *Nucleic Acids Research*, 22:4673–4680.
- Thorne J. L., H. Kishino, and I. S. Painter. 1998. Estimating the rate of evolution of the rate of molecular evolution. *Mol. Biol. Evol.* 15: 1647–1657.
- Tuffley, C. and M. Steel. 1997. Links between maximum likelihood and maximum parsimony under a simple model of site substitution. *Bull. Math. Biol.* 59:581–607.
- Wagner, H.-J. 1981. The minimum number of mutations in an evolutionary network. *J. Theor. Biol.*, 91:621–636.
- Watrous, L. E., and Q. D. Wheeler. 1981. The out-group comparison method of character analysis. *Syst. Zool.*, 30:1–11.
- Weir, B. S. 1990. *Genetic Data Analysis*. Sinauer Associates, Sunderland, Mass.
- Wheeler, Q. D. 1986. Character weighting and cladistic analysis. *Syst. Zool.*, 35:102–109.
- Wheeler, W. C. 1990. Combinatorial weights in phylogenetic analysis: A statistical parsimony procedure. *Cladistics*, 6:269–275.
- Wiley, E. O. 1981. *Phylogenetics. The Theory and Practice of Phylogenetic Systematics*. Wiley, N.Y.
- Williams, P. L., and W. M. Fitch. 1989. Finding the minimum change in a given tree. Pp. 453–470 in B. Fernholm, K. Bremer, and H. Jörnvall (eds.), *The Hierarchy of Life*. Elsevier Press, Amsterdam.
- Williams, P. L., and W. M. Fitch. 1990. Phylogeny determination using dynamically weighted parsimony method. *Methods in Enzymology*, 183:615–626.
- Yang, Z. H. 1993. Maximum-likelihood estimation of phylogeny from DNA sequences when substitution rates differ over sites. *Mol. Biol. Evol.* 10:1396–1401

- Yang, Z. H. 1994a. Maximum-likelihood phylogenetic estimation from DNA sequences with variable rates over sites: Approximate methods. *J. Mol. Evol.* 39:306–314
- Yang, Z. 1994b. Estimating the pattern of nucleotide substitution. *J. Mol. Evol.* 39: 105–111
- Yang, Z., N. Goldman, and A. Friday. 1995a. Maximum-likelihood trees from DNA sequences: A peculiar statistical estimation problem. *Syst. Biol.* 44:384–399
- Yang, Z., S. Kumar, and M. Nei. 1995b. A new method of inference of ancestral nucleotide and amino-acid sequences. *Genetics* 1995: 1641–1650.