PICA

Phylogenetic Inference by Compatibility Analysis

```
    1,1
   /    \
1,0     0,1
   \    /
    0,0
```

Version 4.0

Mark Wilkinson
Department of Zoology
Natural History Museum
London, SW7 5BD
England, U.K.

PREFACE

PICA comprises a set of programs (turbo pascal DOS executables) that implement several probabilistic approaches to compatibility-based phylogenetic analyses. PICA was first developed in the Department of Geology, University of Bristol, while I was a postgraduate student and Research Assistant to Prof. Mike Benton. The absence of programs to implement Sharkey's (1989) method of character ranking led me to program this and some alternative but related approaches and led to a first release of PICA in 1992. Subsequent releases (PICA94, PICA95) added additional programs, with the emphasis shifting to compatibility-based randomisation tests. This release of PICA includes programs implementing methods described in Wilkinson (1997a, 1998a) and all the previous programs have been substantially revised to make them easier to use. Time constraints have prevented me from implementing many of my ambitions for PICA: not all of the programs include as many options as I would have liked, and plans for several additional programs have been temporarily shelved. If you have specific needs that aren't met by the current programs please let me know and I may be able to provide additional software that is in development. I must stress that my programming skills are limited and that the programs are 'no frills' number crunchers. Until that happy day when the programs are rendered redundant by more sophisticated and user-friendly software, the package may be further developed, in which case bug reports, upgrades or notices of upgrades, will be distributed to all registered users. If you did not receive your copy from me then please register.

Restrictions and Citation

PICA 4.0 is distributed for the sole purpose of facilitating and promoting research and is a non-commercial product. The programs and this manual may be freely copied and distributed. The recommended citation is:

> Wilkinson, M. 2001 PICA 4.0: software and documentation. Department of Zoology, The Natural History Museum, London.

Not all of the programs have been extensively tested, particularly with large data sets.

I believe the programs to be free of major bugs.

Please register as a user and report any bugs to me:

e-mail: *marw@nhm.ac.uk* - Tel: +44 (0)20 7942 5164 - Fax: 7942 5433

© Mark Wilkinson, 2001

You should have received the following programs and files:

Program Files:

- ALROY.EXE
- BOILPICA.EXE
- DNALQP.EXE
- FAITH1-4.EXE
- JACTAX.EXE
- LQPROB.EXE
- MATRIX.EXE
- NATRIX.EXE
- PARTPICA.EXE
- RPTP.EXE
- SETPICA.EXE
- SPLIT.EXE

Data and split files:

- EXAMPLE.DAT
- EXAMPLE.SPL

DOS Batch files:

- BOILDOWN.BAT
- FAITH.BAT

Manual:

- PICA4.PDF

INTRODUCTION

Almost always, systematic data includes characters that provide contradictory evidence of phylogenetic relationships. Compatibility analysis includes a number of techniques that make use of the Le Quesne test of character compatibility (Le Quesne, 1969), or analogous compatibility tests, to identify when two characters provide contradictory evidence of phylogenetic relationships. For a pair of characters, each with two alternative character states (e.g. 0 and 1), there are four possible permutations of character states that we can observe in organisms (00, 01, 10, 11). If characters are non-polar, then observation of all four possible permutations means that the characters are mutually incompatible. If each character evolved with only a single character state transformation (so that both fit the true phylogeny), at most three permutations of character states can be produced. Observation of all four permutations implies at least three transformations for the pair of characters, and homoplasy in at least one of the pair of characters. For polarised binary characters, we need only observe all three of the permutations of character states that comprise one or more derived state for the characters to be incompatible. This reflects the fact that the *a priori* hypotheses of polarity of the two characters imply the existence of an ancestral taxon with the combination of both primitive character states (Meacham, 1984). A graphical representation of the Le Quesne test (title page) is used as a motif for this package.

Pairs of characters that are compatible need not be free of homoplasy, and single pairwise Le Quesne tests are of limited value. However, patterns of (in)compatibility in data that are revealed by exhaustive testing of all pairs of characters can identify characters that are particularly responsible for conflict in the data and may therefore be inferred to be the more likely to be homoplastic. For example, consider three pairwise Le Quesne tests of the three characters X, Y and Z. If we find that X and Y, and X and Z are incompatible pairs and that Y and Z are compatible, this entails homoplasy in at least one of X and Y, and at least one of X and Z. The observations *entails* nothing further about the location of the homoplasy, but *suggests* that the homoplasy is in X. This is the simplest (but not the only) explanation and its plausibility increases to the extent that X is incompatible with additional mutually compatible characters.

One of the earliest applications of compatibility methods was in the elimination of conflict by the selection of preferred characters based on repeatable quantitative methods (Le Quesne, 1969; 1972). In particular, clique analysis allowed the identification of maximally large collections of mutually compatible characters (Estabrook *et al*., 1977). These techniques lost popularity in the 1970’s and 1980’s partly as a result of particularly hostile criticisms from advocates of parsimony analyses and from Felsenstein’s (1978) demonstration that both parsimony and clique methods can be positively misleading. Some of these criticisms, such as Felsenstein’s, have retained their force whereas others appear quite unreasonable in retrospect. For example, Kluge’s (1976) criticism that clique analysis may yield multiple optimal trees applies equally to parsimony analysis, although this was not appreciated at the time. Unfortunately, antipathy to clique analysis tends to be extended uncritically to other applications of compatibility methods. In my view, the widespread reluctance to utilise compatibility methods is a clear case of throwing the baby out with the bath water. Other applications of compatibility methods that have been suggested include: to provide information used for ranking or weighting characters (Farris, 1969; Gauld & Underwood, 1986; Sharkey, 1989; Wilkinson, 1992a, 1994a) or otherwise measuring their quality (Wilkinson, 1997a) or using apparent quality in tree selection (Salisbury, 1999), to identify problematic taxa that are particularly responsible for conflict (Guise *et al*., 1982; Gauld & Underwood, 1986), to order multistate characters (Wilkinson, 1992a, 1997b; Sharkey, 1994), as a secondary optimality criterion for choosing among equally parsimonious tees (Sharkey, 1989; Rodrigo, 1992), to provide statistical tests of null hypotheses that individual characters, hypothesised taxa or entire data sets are random/uninformative (Le Quesne, 1989; Wilkinson, 1992a, 1997a,c; Alroy, 1994; Meacham, 1994; Day *et al*., 1998), to explore reticulate evolution (Jakobsen & Easteal, 1996), to test for clonality and recombination in molecular population ecology (Mes, 1998; van der Hulst *et al*. 2000) to test for minimal tree length in parsimony analysis (e.g. Foulds *et al*. 1979, Wilkinson, 1998b), to produce Nelson (Page, 1989) and reduced Adams consensus trees (Wilkinson, 1994b), to build supertrees from matrix representations of trees (Rodgrigo, 1996; Purvis, 1995) and to extract information from partition tables summarising bootstrap analyses (Wilkinson, 1996). An excellent, though no longer comprehensive review of compatibility methods is provided by Meacham & Estabrook (1985). I am convinced that compatibility methods have much to offer systematists, particularly in the realms of data exploration, character weighting, and the statistical testing of data quality, congruence, and hypotheses of relationships.

SOME TERMINOLOGY AND BACKGROUND

In this section I provide some basic terminology, conceptual background and information on the techniques implemented in PICA. The coverage here is in many ways superficial and I stress that it is no substitute for the primary literature, which in most cases describes the concepts, measures and tests in greater detail and provides examples of their use.

There are already a number of synonyms for the tests and measures included in PICA, which may cause some confusion. To promote clarity, particularly with respect to the naming of permutation tail probabilities (PTPs) based on different measures and different randomisation procedures, I have adopted a new scheme of nomenclature that differs from previous terminologies including my own. Most of the programs in PICA allow assessments of (1) entire data matrices, (2) of data partitions, (3) of individual characters, or (4) of particular hypotheses of relationships, using a range of different measures and alternative randomisation procedures. The nomenclature I have adopted is motivated by the desire to make plain the relationship of the test statistic to both what is being assessed and how it is being assessed. Thus names and acronyms begin with in indicator of what is being assessed, followed by one or more indicators of how the assessment is made. The terminology is explained in the text and summarised in Table 1 and the Appendix.

Characters

Characters have a dualistic nature. Operationally, characters are representations of biological variation using discrete partitions (character states) of that underlying variation. Conceptually characters represent hypotheses of homology and therefore of relationships. Thus, in one sense, characters (at least those that are not invariant) are trees (sets of one or more splits). Polar and non-polar characters are equivalent to rooted and non-rooted trees respectively. Unordered multistate and incompletely ordered multistate characters are frequently represented by reticulate character state trees or networks. Here the reticulation is not an hypothesis of relationships as such, but is a graphical representation of several alternative hypotheses of relationships, and thus of uncertainty. Using compatibility methods, characters can be compared to each other and to hypotheses of relationships (splits).

Splits: Full, Partial and Rooted

A *split* is a bipartition of taxa (Bandelt & Dress, 1992; Penny *et al*. 1993). Each branch or edge of a tree partitions the taxa into two mutually exclusive and exhaustive sets (a *full* split). In rooted trees (*rooted* splits) at least one of these sets corresponds to a clade. *Partial* splits are a partitioning of taxa into two mutually exclusive but not exhaustive groups (some subset of taxa are unassigned). Full and Partial splits are somewhat more euphonious terms for the relationships I have denoted as *n*-taxon statements/partitions of varying cardinality (Wilkinson, 1994b, 1995a). A split can be represented as a binary character (Farris, 1973; Baum, 1992; Ragan, 1992) and its compatibility with a set of data can be determined just as if it were a character (Alroy, 1994). Under the view that characters are hypotheses of relationships, characters are splits (but splits need not be characters). In PICA, splits are represented using the '0' and '1' symbols. As a result, full splits are equivalent to binary numbers, and this provides a useful indexing system for splits (Lento *et al*. 1995). For some of the tests in PICA, the size (i.e. the number of taxa) in the subsets defined by a split is important. For rooted splits, split size is taken as the number of taxa in the clade. For unrooted splits, split size is simply the size of the smaller subset of taxa.

(In)compatibility: Conflict, Nesting and Support

Given a set of taxa, two characters *conflict* or are *incompatible* if there exists no non-reticulate tree for these taxa on which both characters can evolve without homoplasy (Meacham & Estabrook, 1985). If characters do not conflict then they are compatible. Viewed as trees, two characters are compatible if they jointly contradict the assumption that they both represent relationships in a single true non-reticulate tree for the taxa. These relations can be determined for pairs of characters, for pairs of splits or for a split and a character. I use *conflict* and *incompatibility* interchangeably, but use only the abbreviation for conflict (**C**) in the naming of randomisation test statistics based on this concept.

In the simple Le Quesne test for binary characters described in the introduction, (in)compatibility can be determined by counting the number of observed permutations of character states and comparing this with the maximum possible if the characters are compatible. This test makes no assumption about character polarity. If characters are polar, this is equivalent to assuming a priori the existence of a common ancestor that has both characters in the assumed primitive state, and this assumption can be incorporated into a polar test.

For some purposes it is useful to distinguish subcategories of the compatibility of characters. A pair of compatible polar binary characters may be compatible through *disjunction, nesting, or equivalence* (Meacham, 1981; Alroy, 1994; Wilkinson, 1998a). They are disjunct if the intersection of the taxa with their derived states is empty. Nesting occurs when the set of taxa with the derived state of one character is a subset of the set of taxa with the derived state of the other character. Nesting (**N**) has also been termed hierarchy (Alroy, 1994) but the former is used here because there are both inclusive and exclusive hierarchies. Equivalence is where the characters have the same taxonomic distributions of derived and primitive character states and thus represent the same hypothesis of relationships. Equivalence might be considered a special case of nesting. Disjunction and nesting are relations that do not apply to non-polar characters, but equivalence does apply. For non-polar characters equivalence is when the characters partition the taxa into identical subsets.

These various sub-types of compatibility give rise to separate measures and tests that capture different aspects of the data. In PICA measures and tests based on the concept of equivalence are used only in the evaluation of splits (hypotheses of relationships). Equivalence between a character and a hypothesised split is termed *support* (i.e. for the split) (**S**) and used in the naming of randomisation test statistics (see below). I have previously used the similar term *equivalent* in a different but related sense that is only likely to cause confusion. Here, the terms equivalence and equivalent are not used in the naming of randomisation test statistics.

Conflict, Nesting and Support of Characters, Matrices, Partitions and Splits

For a given character in a given data set, *character conflict* is simply the total number of characters with which the character conflicts. For a given data set, *matrix conflict* (**MC**) is the total number of pairs of characters that conflict. For a subset of the characters, *within partition conflict* (**WC**) is the number of pairs of characters within the partition that conflict. *Split conflict* (**SC**) is the number of characters with which a specific split is incompatible. *Between partition conflict* (**BC**) is the number of pairwise conflicts between characters drawn from different partitions. In PICA, character and matrix conflicts are typically reported as incompatibilities or incompatibility counts. Analogous nesting and support measures can be defined for characters, matrices, partitions, and splits. Of these, this version of PICA uses *matrix nesting* (**MN**), *within partition nesting* (**WN**), *between partition nesting* (**BN**), *split nesting* (**SN**), and *split support* (**SS**). The precise definition of the latter requires some consideration of weighting and is discussed more fully below.

Ordered and Unordered Multistate Characters

Pairwise tests of character compatibility can be applied to ordered multistate characters. Two ordered multistate characters are compatible *iff* all of their binary factors are compatible. While a pair of ordered multistate characters may be incompatible, some subset of their binary factors may be compatible. In PICA ordered multistate characters must be represented by their binary factors and the focus is upon the compatibilities of these binary factors rather than upon the compatibility of the multistate character as a whole. This has implications for weighting. In PICA all columns in the input data matrix are given equal weight for determining conflict, nesting or support measures (so that weights are not 'scaled'). Thus the n-1 binary factors of a single n-state character each have weight of unity rather than a fractional weight of 1/(n-1). The binary factors of a multistate character are necessarily mutually compatible ($p = 1$). In calculating probabilistic measures, and in randomly permuting data the interdependence of binary factors (or of other characters that are not independent) should not be overlooked and most of the programs in PICA will treat characters as not independent if instructed to by the user (see *Input File Format*, *integer codes*).

Pairwise tests of (in)compatibility can be applied to unordered multistate characters (such as nucleotide sequences). Most programs in PICA can determine compatibilities for unordered characters with up to five character states using a method based on that of Estabrook & Landrum (1975) combined with a simple pre-test that simply compares the number of observed combinations of character states with the maximum number permitted if the characters are compatible. Day *et al.* (1998) distinguish the compatibility of unordered multistate characters as *potential* compatibility because it depends on the existence of compatible orderings of the characters.

Cliques and Clique Analysis

An early focus of compatibility analysis was in finding sets of mutually compatible characters. A set of mutually compatible characters that is not a subset of some larger set of mutually compatible characters is a *clique*. Clique size provides a possible criterion for choosing among alternative phylogenetic hypotheses and breaching the impasse created by incompatible data (Estabrook *et al*. 1977). Computer implementations of clique analysis include CLINCH (Fiala, 1984) and the CLIQUE program in Felsenstein's (1995) PHYLIP package. The method is not featured in PICA.

Probabilities and Expectations

The probability of two characters conflicting given the observed numbers of taxa with each character state if the assignment of character states to taxa is random can be calculated analytically (Le Quesne, 1969; Meacham, 1981). Alternatively, it can be estimated by repeated random permutation (i.e. shuffling) of the assignment of character states to taxa, or determined by exhaustive enumeration of all possible permutations. The probability depends on the numbers of taxa with (or in) each character state for each character. SETPICA.EXE calculates probabilities of compatibility analytically but it can be used only with binary character data. The probability of compatibility for binary factors of multistate characters can be set at unity. Other programs, including all those that accept multistate data, use either random permutation or exhaustive enumeration to investigate probabilities of conflict, nesting or support between characters, or between characters and splits.

Permutations may be of individual characters, of partitions, of entire matrices, or of splits. In PICA, individual character permutation is used only in the determination of Le Quesne probabilities (see below), and their probability of compatibility with other interdependent characters can be set at unity. In randomisation tests of splits probabilities can be determined by randomly permuting the character data or by randomly permuting the hypothesis of relationships represented by the split. Most randomisation tests involve randomisation of the data (single characters or entire matrices). I use *relationship* and the abbreviation (**R**) in the in the names and acronyms of randomisation tests of splits to distinguish those tests that use randomisation of the split rather than randomisation of the data. Similarly, where the randomisation is of the assignment of characters to partitions I use *partition* (**P**) to distinguish these tests. Where the entire character data is randomly permuted, most programs allow collections of interdependent characters to be randomly permuted together so as to preserve their mutual compatibility.

The expected number of conflicts, nestings or support of a character, a matrix, a split, or between character partitions is the arithmetic sum of the corresponding probabilities for all relevant pairwise comparisons. Alternatively, it is the average found by exhaustive enumeration or approximated by random permutation.

Coefficient of Character State Randomness and Normal Deviate

The *Coefficient of Character State Randomness* (CCSR) is the number of observed compatibilities divided by the number expected for random data. Le Quesne (1969) introduced the CCSR, which can be calculated for data matrices or for individual characters, and reported it as a percentage. Here, this simple measure of data quality is reported as a simple ratio. Gauld & Underwood (1986) and Sharkey (1989) called the CCSR the 'randomness ratio'. In PICA characters having no incompatibilities including those that are uninformative under parsimony are assigned a CCSR of zero.

The *normal deviate* (NDEV) is a z-score that describes the difference between the observed and expected number of incompatibilities in units of standard deviations from the mean. Use of the normal deviate in boildowns (see below) is premised upon the normal approximation of the binomial distribution, otherwise it is based upon direct estimation of standard deviations. Le Quesne (1972, 1989) introduced the normal deviate for both individual characters and entire data matrices. It can be used simply as a measure of data quality. As a z-score, the normal deviate can also be interpreted as a test statistic for the null hypothesis that the entire data or an individual character is random and by implication phylogenetically uninformative (Le Quesne, 1989). However, use of the normal deviate as a test of individual characters may be compromised by non random relations among the other characters in the data matrix, which can result in polymodal distributions of the compatibility of random characters (pers. obs.). In PICA, characters that are uninformative under parsimony are assigned a normal deviate of zero.

Boildowns and Absolute Weights

Gauld & Underwood (1986) developed a procedure in which character CCSRs are determined and the character(s) with the worst (highest) CCSR are eliminated, CCSRs are recalculated for the reduced data and the next worst characters eliminated and so on until there is no incompatibility among the remaining characters. This iterative (successive approximations) procedure is similar to that described by Le Quesne (1969, 1972) for several of his non-probabilistic character selection methods and can also be based upon other measures such as the normal deviate or the Le Quesne probability (Wilkinson, 1992a). I use *boildown* as a general term for procedures having this form without regard to the specific measure employed in selecting the 'worst' character(s). BOILDOWN.BAT controls the operation of SETPICA.EXE and BOILPICA.EXE which jointly implement normal deviate and CCSR boildowns (of binary data only).

Boildowns will identify a set of mutually compatible characters that may or may not be a clique. I refer to such sets of characters as *core sets*. We may consider that some characters are particularly important. In that case it would be useful to be able to specify that a core set identified by a boildown should include those characters. PICA allows this through specification of a *core clique constraint*: a listing of characters that should not be eliminated in the boildown no matter what their CCSRs or normal deviates. Note that in this release of PICA characters in the constraint must be mutually compatible.

Sharkey (1989) suggested using the CCSR boildown to provide an absolute ranking of characters. Character rank is determined by the reverse order of their elimination in the boildown. Using Sharkey's method the core clique characters are assigned the highest rank and used to produce a phylogenetic hypothesis. Other lower rank characters are then reintroduced (last out first back in) and their parsimonious interpretation is used to provide additional resolution under the constraint that characters of lower rank can never overturn hypotheses supported by characters of higher rank. Wilkinson (1994a) described how Sharkey's absolute ranking could be represented by an analytically equivalent differential numerical character weighting, termed *absolute weighting*, that could be used to greatly simplify the practical application of Sharkey's method. PICA calculates absolute weights for characters as part of the boildown.

Randomisation Tests and Permutation Tail Probabilities

Randomisation tests of systematic data were developed independently by Archie (1989a) and by Faith & Cranston (1991) in the context of parsimony analysis. Their tests use the lengths of most parsimonious trees (MPTs) as a measure of the hierarchical correlation and possible phylogenetic structure/signal in the data. They compare tree length for the original data to tree lengths for randomly permuted data, in which the assignment of character states to terminal taxa are randomly permuted within characters while preserving the observed numbers of taxa with (or in) each character state. These tests are aimed at evaluating the null hypothesis that a data matrix is indistinguishable from randomly permuted and thus phylogenetically meaningless data in which the congruence among characters (as measured by tree length) is no greater than would be expected by chance alone. Faith & Cranston's (1991) parsimony permutation tail probability (PTP) is the proportion of tree lengths as short or shorter than that for the original data for all data sets. Because the original data set is included in the sample, the minimum PTP is $1/(N+1)$ where N is the number of randomly permuted data sets. Parsimony PTPs below an accepted cut-off (e.g. 0.05) allow rejection of the null hypothesis that the data has no more congruence (structure) than expected by chance alone. Failure to pass the test suggests that phylogenetic inferences based on the data have little or no credibility and that the data is not suitable for phylogeny reconstruction. Other parsimony PTPs have been developed. For example, Faith (1991) introduced tests for hypotheses of monophyly and non-monophyly that are based on length differences between MPTs that differ with respect to some group. Phylogenetic randomisation tests have proven controversial. However, more generally, randomisation tests are widely used and respectable statistical methods (see, Manly [1991] for an authoritative introduction) and I see no reason why they should not be used in a phylogenetic context.

Randomisation tests and PTPs can be based on other measures of data quality. Archie (1989a) suggested that randomisation tests could be extended to clique analysis by testing the informativeness of characters that are excluded from preferred cliques. To my knowledge this interesting suggestion has not been explored further, but other compatibility-based applications of randomisation tests have been proposed. A CSPTP (clique size PTP) test directly analogous to the original parsimony PTP was developed by Wilkinson (1992a,b). The CSPTP is the proportion of sampled data sets having maximal cliques as large or larger than the original data. This test is not included in this version of PICA. The CSPTP test is based, like parsimony-based tests, on a measure of hierarchical structure that is also a criterion for choosing among phylogenetic hypotheses. Similar tests could be developed for any method where tree choice is based on some optimality criterion. The performance of different methods in randomisation tests might sometimes reveal that particular methods are better able to discriminate signal from noise in particular data sets with obvious implications for choice of method (Archie, 1989a; Wilkinson, 1992a).

Alternative compatibility randomisation tests have been developed that use compatibility relations among characters or between splits and characters as measures of data quality or of the quality of particular phylogenetic hypotheses. These can be calculated relatively easily without the need to compute or choose a tree, and they providing a method- or tree-independent set of randomisation tests.

Table 1. Summary of characteristics of randomisation tests and associated abbreviations.

Property			
General	***Specific***	***Abbreviation***	***Synonyms***
Focus	***Matrix***	***M***	***-***
	Within Partition	***W***	
	Between partitions	***B***	***-***
	Character	***none used***	***-***
	Split	***S***	***-***
Measure	***Conflict***	***C***	***Incompatibility***
	Nesting	***N***	***Hierarchy***
	Support	***S***	***Equivalence***
Random Permutation	***Data (all characters)***	***none used***	***-***
	Partitions	***P***	***-***
	Single character	***none used***	
	Relationship	***R***	***Equivalent***

Table 1, summarises the different properties of the PTP tests in PICA. As noted above, tests differ with respect to their focus (individual characters, matrices, partitions, and splits), the measure used (conflict, nesting, support) and in the kind of randomisation. These properties are combined in this order (focus, measure and, where necessary, randomisation procedure) to give the name of the PTP test statistic. The kind of randomisation procedure is not indicated if it is the character data that is randomly permuted (i.e. the assignment of taxa to character states). Randomisations of the assignments of characters to partitions or of relationships are distinguished as **PPTP**s and **RPTP**s respectively. Each property referred to in the name of a test statistic is represented by a single letter abbreviation giving each test a unique acronym. For example, the matrix conflict PTP (MCPTP) is a test of a matrix using character conflicts and character randomisation, and the BCPPTP is a test of the conflict between partitions using randomisation of partitions. The names and acronyms of the PTP tests as used in PICA and their synonyms in other works are given in the Appendix.

That a data set passes a matrix randomisation test does not justify the conclusion drawn by some that the data contains ‘significant phylogenetic structure’. The structure it contains may or may not be phylogenetic. Furthermore, it does not indicate the location of the structure. With most matrix randomisation tests, it is likely that relatively little potentially phylogenetic structure is needed to reject the null hypothesis and that data sets that pass the test could include considerable random data. Faith & Cranston (1991) described conditional parsimony PTP tests in which only a subset of the data set was randomised. Most programs in PICA allows such conditional tests through specification of taxa (e.g. ingroup), and in some cases subsets of characters, to be excluded from the random permutation.

Matrix Conflict and Matrix Nesting PTPs

Wilkinson (1992a,b), Alroy (1994) and Day *et al.* (1998) independently developed a randomisation test of data matrices that uses the pairwise (in)compatibilities of all characters within the matrix as a measure of the quality or strength of phylogenetic signal in the matrix. The greater the conflict the lower the quality of the data. The test statistic, is here termed the matrix conflict PTP (MCPTP) = the proportion of data sets (original and randomly permuted) having conflict (i.e. an incompatibility count) as low as, or lower than, the original data. Alroy (1994) described an analogous tests using nesting (hierarchy), a subset of the relations that can pertain between compatible characters (see also Wilkinson, 1998a). The matrix nesting PTP (MNPTP) = the proportion of data sets (original and randomly permuted) that have as much or more nesting pairs than the original data. These tests are implemented in MATRIX.EXE and NATRIX.EXE respectively. Alroy (1994) considered the nesting test more sensitive to phylogenetic, as opposed to other correlations between characters, in a data set than the MCPTP test. However, it is applicable only to polar characters and, as implemented in PICA, only to binary characters.

These matrix PTPs describe the proportions of randomly permuted data sets that, judged in terms of conflict or nesting, are of as good or better quality than the original data. If this proportion is insufficiently small then we cannot reject the null hypothesis that the original data is not significantly better than randomly permuted data. Given that the success of all phylogenetic methods depends upon data quality, failure to reject the null hypothesis implies that the data should be judged unreliable for phylogenetic inference. Passing matrix randomisation tests might reasonably be considered a minimum requirement of phylogenetic data. What is considered significant is, of course, decided by the investigator. The test here is one-tailed, so the ‘standard’ level of significance is when the PTP is ≤ 0.05.

Within and Between Partition Conflict PPTPs.

Parsimony-based incongruence or partition homogeneity tests have been developed for assessing the significance of incongruence between data partitions (Farris *et al*. 1994; Huelsenbeck *et al*. 1996). Analogous compatibility-based tests of partition heterogeneity compare the level of *between partition* incompatibility to that expected for random partitions of the same size (Wilkinson, 1997a). The greater the conflict the more heterogeneous the partitions. Random partitions represent a standard for ‘non-significant’ heterogeneity against which the original partitions are judged. In this test it is the assignment of characters to partitions that is randomly permuted rather than the data and this yields *partition* (P)PTPs. The between partition compatibility PPTP (BCPPTP) = the proportion of partitions (original and randomly permuted) for which the between partition conflict (incompatibility) is as high or higher than for the original partitions. The BCPPTP is a test statistic for the null hypothesis that the original partitions are no more heterogeneous than random partitions. The test is mostly used in determining whether data should or should not be combined, with significant heterogeneity being taken as grounds for not combining data. With molecular data, heterogeneity between genes may be indicative of different gene trees. The test can also be used to explore potential interdependencies between characters and possible correlated homoplasy (e.g. Wilkinson, 1997a).

The conflict within each partition can also be compared to that for equivalent sized but randomly permuted partitions in order to determine if partitions are of significantly high or low quality compared to the ‘standard’ given by a random partition of the same size. The within partition conflict PPTP (WCPPTP) = the proportion of partitions (original and randomly permuted) for which the within partition conflict (incompatibility) is as low or lower than for the original partition. The WCPPTP can be used as a one tailed test if a specific hypothesis is being tested but can otherwise be used as a two-tailed test. A low (e.g. ≤ 0.025) WCPPTP indicates significantly little conflict, and a high (e.g. ≥ 0.975) WCPPTP indicates significantly high conflict, in the partition. Comparison of results for different partitions can be used to rank or otherwise compare the data quality of partitions. These PPTP tests are implemented in PARTPICA.EXE. A limitation of the current implementation is that it ignores any independence specified for the character data.

Within and Between Partition Conflict and Nesting PTPs.

Within and between partition conflicts can also be compared to randomly permuted data producing standard PTPs. The within partition conflict PTP (WCPTP) = the proportion of data sets (original and randomly permuted) for which the within partition conflict (incompatibility) is as low or lower than for the original data. The test is essentially the MCPTP test except that the focus is on the partition rather than the entire matrix. Similarly a within partition nesting PTP (WNPTP) = the proportion of data sets (original and randomly permuted) for which the within partition nesting is as high or higher than for the original data, and this test is similarly related to the corresponding MNPTP.

The between partition conflict PTP (BCPTP) = the proportion of data sets (original and randomly permuted) for which the between partition conflict (incompatibility) is as low or lower than for the original data. Similarly the between partition nesting PTP (BNPTP) = the proportion of data sets (original and randomly permuted) for which the between partition nesting is as high or higher than for the original data. The interpretation of the BCPTP depends on the details of the randomisation. In PICA these statistics can be calculated for no more than a pair of partitions. Where the characters in both partitions are randomised the expectation is that the conflict between phylogenetically informative partitions should be less, and nesting greater, than for similar partitions of randomly permuted and phylogenetically uninformative data. A significantly low BCPTP or BNPTP thus allows the rejection of the null hypothesis that the partitions show no less conflict or no more nesting than random data. Because random data rather than random partitions provide the basis of the comparison, these are likely to be much weaker tests than the corresponding PPTP tests. PICA also allows conditional tests in which the data in only one of the partitions is randomised. In this case, the BCPTP and BNPTP are test statistics for the null hypotheses that the randomised partition is random with respect to the partition that is held constant. These tests are implemented in NATRIX.EXE and MATRIX.EXE.

Tests of Splits: A Priori, A Posteriori, and Relationship

Faith (1991) introduced parsimony-based randomisation tests of hypotheses of monophyly and non-monophyly. Tree-independent tests of hypothesised relationships have been developed using compatibility. Alroy (1994) introduced the promising idea of using the compatibility and nesting counts for hypothesised splits as measures of the ‘support’ for the splits provided by the data and as the bases for randomisation tests of splits. Wilkinson (1998a) introduced an alternative measure of split support and associated randomisation tests. All of these tests involve randomisation of the character data. Faith (1991) distinguished between *a priori* and *a posteriori* parsimony randomisation tests of hypothesised relationships. According to Faith, a priori tests are appropriate for testing hypotheses that have been forwarded independently of the analysis of the data to hand. In contrast, a posteriori tests are appropriate for testing relationships that are of interest because they are supported by the data. This distinction has been extended to compatibility-based tests (Wilkinson, 1998a) and PICA implements both a priori and a posteriori tests of hypothesised relationships through randomisation of the data. In addition, PICA includes *relationship* (R)PTP tests in which the relationship (i.e. a split) is randomised rather than the data (Wilkinson, 1998a; Frohlich & Estabrook, 2000).

Split Conflict and Nesting PTPs

Alroy (1994) used compatibility count for a split as a measure of the ‘support’ provided by the data for the hypothesised split. The number of compatibilities, expressed as a proportion of the number of characters, is the complement of the number of incompatibilities. The latter has been referred to as split conflict by Penny and his co-workers (e.g. Lento *et al*, 1995) and I follow their terminology here. I refer to my equivalent of Alroy's test statistic as the a priori split conflict PTP (SCPTP), which for a split X that has X_{sc} conflict with the original data, = the proportion of data sets (original and randomly permuted) that have as little or less conflict with X than X_{sc}. The corresponding a posteriori SCPTP = the proportion of data sets (original and randomly permuted) in which any supported split of the same size as X has conflict less than or equal to X_{sc}. Although SCPTPs were proposed as tests of the null hypotheses that particular relationships are no better supported than might be expected by chance alone, the use of split conflict as a measure of support is problematic. Wilkinson (1998a) showed that, used in this way, split conflict randomisation tests are very liberal and that their power is dependent upon split size. He also suggested that, because split conflict was a measure of support for incompatible groups and therefore only an indirect measure of support for a particular group, the SCPTP might be better taken as a test of non-monophyly (the strength of evidence against a group) than of monophyly. Used in this way high SCPTPs (e.g. $\geq$ 0.95) indicate significant evidence for non-monophyly. A priori and a posteriori SCPTP tests are implemented by SPLIT.EXE and by four programs (FAITH1-4.EXE) controlled by FAITH.BAT respectively. Both tests can be applied to full splits (bipartitions, components, clades) but partial splits, can only be investigated with a priori tests.

Alroy (1994) also described a test of splits using nesting rather than conflict or compatibility. The split nesting PTP (SNPTP) of a split is the proportion of data sets (original and randomly permuted) with which the split has as many or more hierarchical nestings as it does with the original data. This a priori

PTP test is implemented by ALROY.EXE. An a posteriori test is also readily defined but awaits implementation. This approach is applicable only to polar characters and, in PICA, only binary characters.

Split Support PTPs

Randomisation tests of splits based on split support are a logical extension to Alroy's (1994) tests (Wilkinson, 1998a). Split support is a weighted sum (see below) of the number of characters in the data supporting a particular split. In PICA a character is taken as supporting a split when there are no character states that are common to both the groups of taxa identified by the split. Like hierarchical nesting, the relationships of split support is a subset of the relationships of compatibility, but it differs in its applicability to unrooted trees and to non-polar characters. As with SCPTPs, SSPTPs can be determined for full or for partial splits and they comes in a priori and a posteriori versions that are implemented by SPLIT.EXE and FAITH.BAT respectively (for formal definitions substitute 'support' for 'conflict' and 'as much or more' for 'as little or less ' in the definitions of the SCPTPs given above. Frohlich & Estabrook (2000) referred to this concept of support as '*Wilkinson Support*' to distinguish it from parsimony based concepts. Importantly, a character can provide Wilkinson support for a split when it is parsimony uninformative if it is possible for non-homoplastic character state changes to have occurred on the branch corresponding to the split (see Wilkinson [1998a] and Frohlich & Estabrook [2000] for further discussion and explanation). As used in PICA, split uninformative characters are those that cannot support any non-trivial split (i.e. of size greater than one).

PICA uses a weighted split support measure. Complete binary characters support a single split, but with unordered multistate data, or where there are missing entries, a given character may be interpreted as supporting several splits. Importantly the number of splits a character supports is a function of the numbers of character states, and increases faster than the number of splits that can be supported simultaneously by any single set of non-homoplastic (i.e. parsimonious) character state changes . A weighted split support measure is provided by downweighting the split support provided by each character to each split according to the ratio of the number of splits it can simultaneously support to the number of splits it can potentially support (Wilkinson, 1998a). It should be noted that although the underlying support concept is the same, the weighting of split support used here is very different from the measure of Wilkinson support employed by Frohlich & Estabrook (2000).

Split support randomisation tests are a direct means of testing the level of support for phylogenetic relationships without building trees. Passing such tests might reasonably be considered a requirement for relationships to be considered well supported by the data, and building phylogenetic trees from all splits that pass am posteriori support randomisation tests (a reduced spectrum of splits in the data) using either compatibility of parsimony methods with support based weighting is an interesting option (Wilkinson, 1998a). An important caveat is that the tests are sensitive to taxonomic scope (i.e. the number of taxa included). This warrants further study and suggests that, with large numbers of taxa, an experimental approach in which subsets of the data are investigated should be adopted.

Split Support and Split Conflict Relationships PTPs

The a priori and a posteriori split support, nesting and conflict PTPs described above are based on comparison of these measures from the original data with the scores they achieve in random permutations of that data. Wilkinson (1998a) pointed out that the relationship between data and splits can be reduced to that expected by chance alone either by random permutation of the data or by random permutation of the split. The latter procedure involves randomly permuting the membership of taxa across a split to producing a sample of random splits each of the same size as the original. Frohlich & Estabrook (2000) also used random permutation of either splits or of data in their Wilkinson Support Analyses. Permutation tail probabilities determined from comparison of the fit of the data to a given split to the fit achieved by comparable random splits are distinguished as relationship PTPs (RPTPs). Thus, whereas the SSPTP is a test statistic for the null hypothesis that the split has no more support from the original data than with random and phylogenetically uninformative data, the SSRPTP (split support RPTP) is a test statistic for the null hypothesis that a split has no more support from the data than do comparable (same sized) but random splits. The difference is important. PTP tests based on data randomisation represent a kind of worst case scenario of data quality. In contrast, RPTP tests make no assumptions about the data, other than it is as it is. For data sets with few taxa, the number of permutations of splits may be low and this should be taken into account when considering appropriate levels of significance. In PICA, RPTP tests are implemented for support and conflict measures in RPTP.EXE. Wilkinson (1998a) suggested that restricting the comparison of splits scores with those of only incompatible randomly permuted splits might be justified on the view that the scores of compatible splits are not relevant. This option, which should produce a more conservative test, is implemented in RPTP.EXE.

Le Quesne Probability

The Le Quesne probability (Wilkinson, 1992a) is a test statistic for the null hypothesis that a particular character is no less incompatible with the other characters in the data than is a random character. Failure to reject the null hypothesis may be taken as grounds for discarding or otherwise downweighting characters. It is determined by random permutation or exhaustive enumeration of the assignment of character states to taxa. The incompatibility count of each permutation is determined and compared to that of the original character. The Le Quesne probability is the permutation tail probability that describes the chance of random permutations of the assignment of character states to taxa achieving as low or lower incompatibility count than the original character. The advantage of this test is that, unlike the character normal deviate, no assumption is made concerning the randomness of the other characters in the data. The only assumption is that these characters are as they are, surely the safest of all possible assumptions. Meacham (1994) has independently developed a measure which is, except for trivial differences, the same as the Le Quesne probability. Meacham (1994) used character weighting based on Le Quesne probabilities to study angiosperm phylogeny. Other examples of the application of the Le Quesne probability are provided by Wilkinson (1997a) and by Wilkinson & Nussbaum (1996). For data sets with few taxa, the number of permutations of a character may be low and this should be taken into account when considering appropriate levels of significance. Le Quesne probabilities can be determined using LQPROB.EXE (binary data only) and DNALQP.EXE.

Jackknife Compatibility

Gauld & Underwood (1986), following ideas first presented by Guise *et al*. (1982), developed a 'marking procedure' to identify when specified taxa were particularly responsible for incompatibilities. In PICA, measures based on this idea are calculated by taxon jackknifing. If we determine the matrix incompatibility count we can compare this to the incompatibility counts for matrices lacking a particular taxon, the difference giving a measure of the number of incompatibilities that do not pertain when the taxon is not considered, and that are therefore in some sense caused by the excluded taxon. JACTAX.EXE calculates the conflicts caused by a given taxon through a first order jackknife (removal of single taxa) or optionally a second order jackknife (removal of all pairs of taxa including the taxon of interest). For the second order jackknife, scores for individual taxa are average conflicts caused by all pairs including the taxa.

JACTAX.EXE allows the number of conflicts caused by particular taxa (first order jackknifing only) to be compared to the expectations under two alternative models by repeatedly replacing the taxon with a randomly generated taxon. In the first model, character states for the random taxon are chosen equiprobably from the observed set of character states for each character. In the second model, character states are chosen with probabilities based on their observed frequencies. Randomly generated taxa will tend to cause fewer conflicts under the second model. With both models, missing entries are maintained as in the original taxon. Tail probabilities based on the comparison of the conflicts caused by the original and the randomly generated taxa serve as test statistics for the null hypotheses that the taxon causes no less conflict than a randomly generated taxon under the specific model. The data for taxa that fail the test might be considered no better than random, but the tail probabilities should be interpreted cautiously. The number of conflicts caused by a taxon depends upon how many unique combinations of character states are represented by the taxon, and how many of these result in additional incompatibilities. Thus a taxon's performance will be highly contingent on what other taxa are included. Similar taxa will tend to cause few incompatibilities in a first order jackknife. However, JACTAX.EXE may help to identify problematic taxa. A corresponding parsimony-based test could be constructed using length differences of tees that include or exclude taxa.

Incompatibility Excess Ratios

In the context of parsimony, Archie (1989b) proposed a descriptive statistic of data quality, the homoplasy excess ratio that is based on comparison of parsimony tree length for the original data and the average length of parsimony trees for randomly permuted data. Analogous measures using incompatibility also describe the extent to which the data departs from the worst case scenario of randomly permuted data (Wilkinson, 1997c). MATRIX.EXE calculates two incompatibility excess ratios (IER_1 and IER_2). $IER_1 = (R-O)/R$ and $IER_2 = (N-O)/N$ where R is the average conflict for randomly permuted data, O is the observed conflict for the original data and N is the ninety-five percent cutoff (the maximum level of conflict that the original data can have while passing the test at the standard level of significance). The latter measure follows a suggestion by Faith & Cranston (1991).

Missing Data

Missing data complicates the calculation of probabilities of (in)compatibility because there are several possible treatments. Using the analytical approach, comparisons involving missing entries (e.g. a '1' in one character is paired with a '?' in another character) can either be ignored (Le Quesne, 1972) which effectively assumes the association is constant, or alternatively the distribution of missing entries, and thus their associations between taxa and characters, can also be considered random (Wilkinson, 1992a). The latter approach is more complex, and SETPICA.EXE implements Le Quesne's simpler alternative. For many of the programs that use random permutation, missing data can be maintained in its original positions or included in the random permutation. Inasmuch as some distributions of missing data (e.g. its concentration in particular taxa) may affect the number of pairwise comparisons that are invalidated by missing entries for one or both of the characters, the former may be preferable because it makes no unnecessary assumptions about the probability of alternative distributions of missing data. For the a posteriori SC and SSPTP tests, splits have to be identified from the original and from randomly permuted data. In the current implementation of these tests only full splits are identified and only from full characters (those with no missing entries). In calculating split support, the presence of missing entries for a character increases the number of splits it can support and decreases the weighted support given to any particular split. The amount of missing data can have a major impact on the conflict caused by individual taxa, typically the more missing data, the less conflict. JACTAX.EXE reports the percentage of missing entries for taxa allowing some monitoring of its effect, but no attempt has been made to standardise conflict scores for levels of missing data in this version of PICA.

INPUT FILE FORMAT

Input files are of two types. ***Data*** files provide listings of taxa and characters and may contain other information used in the analyses. ***Split*** files contain listings of splits for use in a priori and relationship split tests. PICA has its own format requirements for these files. The format is not complicated and data files for use with other phylogeny programs can be converted to PICA format with a little text editing.

DATA FILES

Each data file comprises a first line of information, a block of taxon names and character data, and optional commands and listings of interdependent characters, data partitions, and character selections. Format differs slightly for binary and multistate data. An example data file based on the data for the families of caecilian amphibians of Duellman &Trueb (1986) is given in Box 1 and included in the EXAMPLE.DAT file.

BOX 1. An example data file.

```
7 19 2 0 1 Duellman and Trueb (1986) caecilian data
Hyp-ancestor*    00000 00000 00000 0000
Rhinatrematidae  00000 00010 00000 0000
Ichthyophiidae   01010 00001 10001 1111
Uraeotyphlidae   01110 10001 10101 1111
Scolecomorphidae 11111 10000 01?11 1111
Typhlonectidae   11111 11111 01111 1111
Caeciliidae      11111 11111 01111 1111
codes
1 2 2 3 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17
parts
1111111111222222222
const
2 1 7
```

The first Line of a data file must begin with two numbers, the **number of taxa** and the **number of characters** in that order, separated by at least one blank (e.g. ‘7 19’ in BOX 1, line 1). This is minimum requirement for all programs. After the numbers of taxa and characters, a **symbols statement** can be included in the first line. The symbols statement comprises: the **number of character states** present in the data, which should not exceed 5, and a list of these **character states** all separated by blanks (e.g. ‘2 0 1’, meaning 2 symbols – 0 and 1, in BOX 1, line 1). The symbols statement is ignored by the programs for the analysis of binary data. It is used by the programs for the analysis of multistate data whether the data is multistate or not. If no symbols statement is included the programs for multistate data have a case sensitive **default** of five states (ACGT-). Other symbols except the dot '.', are interpreted by the multistate programs as missing entries. Following the symbols statement **other information** you want echoed to output can be included on the first line after the number of characters and symbols statement if included and separated from them by at least one blank space.

The Data Block occupies subsequent lines and comprises **taxon labels** and the **character data**. Each taxon must be represented by a label that contains no blanks. Each name must be separated from the character data by at least one blank or by a line break. Character data follows the taxon name in free format with blanks and line breaks ignored. Interleaved format is not supported. Unfortunately, PICA 4.0 does not support polytypic terminal taxa or nucleotide ambiguity codes.

Integer Codes provide a means of specifying interdependence between characters. If two or more characters are the *binary factors* of a single multistate character then they will necessarily be compatible. It would be wrong to calculate the probability of these binary factors being compatible at random as if they were independent characters. To specify interdependence each character is assigned an integer. The same integer denotes that characters are not independent. Integer codes are signalled by the command **codes** in lower case beginning on a new line after the character data. The integer codes begin on a new line after the **codes** command and can occupy any numbers of lines of unspecified length, but they must be separated by at least one space or a line break. For example, in BOX 1, the integer codes indicate that there are two pairs of interdependent characters (2 and 3, and 4 and 5). These are the binary factors of ordered multistate characters. Characters do not need to be contiguous (i.e. a single block in the data) to share the same integer code, making this coding useful for representing non-independence of base pairing nucleotides in ribosomal stem structures. If characters are all independent no integer codes are needed. I believe that PICA is the only package with randomisation procedures that incorporate this important feature.

Partition Codes can be signalled analogously by the **parts** command followed by a listing of the partitions to which each character belongs indicated by the symbols 1-2 (or 1-5 for use with PARTPICA.EXE). For example, the data in BOX 1 has two partitions comprising characters 1-10 and 11-19 respectively. Characters need not be contiguous to be included in the same partition. Partitions can be specified to implement conditional randomisation tests and are needed for tests of partitions.

Constraint Codes allow characters that are not to be excluded during a boildown to be identified. They are signalled by the **const** command followed by the number of characters in the constraint and the reference numbers of these characters. For example, the data in BOX 1 has a constraint of two characters, namely character 2 and character 7. Multiple character is included in a constraint should be mutually compatible.

Binary Data: LQPROB.EXE, SETPICA.EXE, NATRIX.EXE and ALROY.EXE work only with **binary data**. The character data for these programs must be represented by the **symbols** '0' and '1', with missing entries represented by the '?' symbol. Some of these programs interpret ‘1’ as the derived condition when polarity is assumed. Other character states will be ignored (treated as missing entries) by the programs.

Reserved symbols and special features: The programs that accept multistate data interpret the **dot** ('.') symbol in character data as indicating identity with the character state of the first taxon in the matrix. The **asterisk** ('*') symbol in a taxon label (which should not be separated from the label by blanks) is interpreted by the programs as an instruction not to include that taxon in the random permutation. This is useful for performing conditional randomisation tests (e.g. ones focusing on an ingroup and holding outgroup taxa constant) or for specifying polarity. When the first taxon in the data has an asterisk in its name (as in the data in BOX 1) then it can be used in some programs to specify character polarity.

Limitations: The sizes of data matrices that can be analysed differ among the programs. If the limits cause you problems you should contact the author for advice.

SPLIT FILES

Splits to be tested with SPLIT.EXE or EQSPLIT.EXE must be specified by a split file. An example split file is given in BOX 2, and included as the EXAMPLE.SPL file.

BOX 2. An example split file.

```
3 caecilian splits
0001111
00?1111
0101010
```

The first line of a split file must begin with the **number of splits** included in the file, separated from any other information on the first line of the file by a blank. Other information will be written to the output file. Splits are represented on subsequent lines by strings of '0's and '1's, and in the case of partial splits by '?'s. The '0's and '1' represent the taxa included in the two groups specified by the split. Their order in the string representing the order of taxa in the data file. Thus, in the example (BOX 3) there are three splits. The first is a full split that divides the taxa into the hypothetical ancestor, Rhinatrematidae and Ichthyophiidae in one group and the Uraeotyphlidae, Scolecomorphidae, Caeciliidae and Typhlonectidae in the other. The second is the similar partial split that excludes the Uraeotyphlidae. The third split was formed following an arbitrary rule of placing adjacent taxa in the data matrix in different subsets of the split. This split is almost certainly random with respect to phylogeny.

THE PROGRAMS

The programs are described below briefly and illustrated with output for the example data in BOX 1 and, where applicable the example split file in BOX 2. This is not the most comprehensive data available for caecilians, and I use it to illustrate the use of the programs rather than as an exercise in inferring or assessing caecilian family interrelationships.

Output from the various programs begins with two lines identifying the analysis used and the date and time of execution. This is followed by an 'ANALYSIS' section giving self explanatory information on the files analysed. The next section, 'SETTINGS' gives information on the analytical options employed. The final section 'RESULTS' summarises the results. The results are best viewed in a simple text editor such as *notepad* or *wordpad*. If viewed in a word processor they will be more readable if you use a proportional font , such as `courier`.

The programs are of four basic types:

1. those that evaluate entire data sets or data set partitions (MATRIX.EXE, NATRIX.EXE, PARTPICA.EXE).

2. those that evaluate individual characters (LQPROB.EXE, DNALQP.EXE and programs invoked by BOILDOWN.BAT).

3. those that evaluate splits (SPLIT.EXE, RPTP.EXE, ALROY.EXE and programs invoked by FAITH.BAT

4. those that evaluate taxa (JACTAX.EXE)

Programs for Evaluating Data Matrices and Partitions

MATRIX.EXE calculates conflict PTPs for entire data matrices (MCPTPs) or for partitions by random permutation of character data. It also reports the distribution of incompatibility counts for the randomly permuted data, and calculates their mean, standard deviation, corresponding normal deviate, 95% cutoff and two incompatibility excess ratios. Where there are two partitions in the data, MATRIX.EXE optionally reports WCPTPs and BCPTPs with options to randomly permute character data in one or in both partitions..

An example output file from an analysis of the data in BOX 1 is shown opposite.

This analysis used 99 random permutations (giving minimum PTPs of 0.01). Note also that the power of the test could be increased by using more random permutations.

Character interdependence, as specified by the integer codes in the data file (BOX 1), was recognised.

Taxon 1 was not included in the random permutation, and missing data were maintained in their original positions. In addition, characters in partition 1 were held constant.

In this case, MATRIX.EXE gives the WCPTP for partition 2 and the BCPTP for the comparison between partitions 1 and 2.

Holding taxon 1 produces a conditional test of the ingroup data, equivalent to assuming polarities given by the hypothetical ancestor.

```
PICA 4.0 - 17/1/2001 17:32
MATRIX   - Matrix conflict randomisation test

*** ANALYSIS ***

example.dat 7 taxa 19 characters
 Duellman and Trueb (1986) caecilian data

*** SETTINGS ***

PTPs approximated by 99 random data permutations
Character interdependence as specified by integer codes
Taxa held constant: 1
Missing entries held constant
Partition 1 held constant

*** RESULTS ***

Partition 2

              RANDOMLY PERMUTED DATA

      Frequencies of Incompatibility Counts

12:1 13:1 15:1 16:1 17:1 18:8 19:9 20:16 21:12 22:19
23:14 24:9 25:7

WCPTP =     0.01
Incompatibility count for partition 2   =     1
Mean count for randomised data          =    21.121 +/- 2.487
Normal Deviate                          =     8.089
95% cutoff                              =    18
Incompatibility Excess Ratios (1)       =     0.953
                              (2)       =     0.944

Between partitions

              RANDOMLY PERMUTED DATA

      Frequencies of Incompatibility Counts

37:2 40:2 41:1 42:3 43:1 45:1 46:4 48:6 49:4 50:6
51:3 52:3 53:7 54:3 55:2 56:4 57:5 58:2 59:4 60:8
61:2 62:4 63:2 64:1 65:4 66:1 67:1 68:2 69:1 70:1
72:1 73:1 74:4 76:1 79:1 80:1

BCPTP =     0.01
Between partition incompatibility count =    12
Mean count for randomised data          =    56.303 +/- 9.429
Normal Deviate                          =     4.699
95% cutoff                              =    42
Incompatibility Excess Ratios (1)       =     0.787
                              (2)       =     0.714
```

The significant BCPTP means we can reject the hypothesis that the character data in partition 2 are random with respect to partition 1.

MATRIX.EXE accepts up to 100 taxa and 500 characters with up to five character states. It accepts character interdependence as specified by integer codes, allows missing data and individual taxa to be held constant or randomised, and offers a set of options for the treatment of two data partitions. It ignores constraints in the data file.

Programs for Evaluating Data Matrices and Partitions

NATRIX.EXE calculates MNPTPs for entire data matrices (polar binary data only) by random permutation of character data. It also reports the distribution of nesting counts for the randomly permuted data, and calculates their mean, standard deviation, normal deviate and 95% cutoff. With partitions, NATRIX.EXE optionally reports WNPTPs and BNPTPs with options as in MATRIX.EXE.

An example output file from an analysis of the data in BOX 1 is shown opposite.

Settings are mostly as for the MATRIX.EXE analysis shown above.

The partition included in the data file was ignored.

The MNPTP is 0.01: we can reject the null hypotheses that the data are indistinguishable from random.

Random data had between 49 and 75 nestings, with an average of 57.162 and standard deviation of 5.079, compared to the 138 for the original data.

```
PICA 4.0 - 17/1/2001 21:21
NATRIX   - Matrix nesting randomisation test

*** ANALYSIS ***

example.dat 7 taxa 19 characters
 2 0 1 Duellman and Trueb (1986) caecilian data

*** SETTINGS ***

PTPs approximated by 99 random data permutations
Character interdependence as specified by integer codes
Taxa held constant: 1
Missing entries held constant
Partitions ignored

*** RESULTS ***

Full data

              RANDOMLY PERMUTED DATA

           Frequencies of Nesting Counts

75:1 71:1 68:2 66:1 65:3 64:2 63:3 62:4 61:10 60:6
59:5 58:5 57:7 56:8 55:7 54:1 53:10 52:14 51:5 50:1
49:3

MNPTP =     0.01
Nesting count for original data      =   138
Mean count for randomised data       =    57.162 +/- 5.079
Normal Deviate                       =    15.917
95% cutoff                           =    65
```

NATRIX.EXE accepts up to 75 taxa and 750 characters. Characters must be binary and polar with ‘0’ and ‘1’ representing the primitive and derived states respectively. It accepts character interdependence as specified by integer codes, allows missing data and individual taxa to be held constant or randomised, and offers a set of options for the treatment of two data partitions. It ignores constraints in the data file.

Programs for Evaluating Data Matrices and Partitions

PARTPICA.EXE calculates PPTPs for up to five character partitions of entire data matrices by random permutation of the assignment of characters to partitions. WCPPTPs are given for each partition and a BCPPTP is given for the overall between partition conflict. It also reports the corresponding observed incompatibility counts and means for randomly partitioned data.

An example output file from an analysis of the data in BOX 1 is shown opposite.

Settings are mostly as for the MATRIX.EXE analysis shown above.

In the example the WCPPTPs are not significant indicating that the partitions are not distinguishable from random partitions.

```
PICA 4.0 - 17/1/2001 17:3
PARTPICA – Within and between partition conflict randomisation test

*** ANALYSIS ***

example.dat 7 taxa 19 characters
 Duellman and Trueb (1986) caecilian data

*** SETTINGS ***

PPTPs approximated by 99 random partition permutations

*** RESULTS ***

Incompatibility count for original data =    24

Partition 1 – WCPPTP =    0.75
              Incompatibility count      =    11
              Mean for random partitions =     6.707

Partition 2 – WCPPTP =    0.11
              Incompatibility count      =     1
              Mean for random partitions =     4.545

Between partitions – BCPPTP =     0.83
              Incompatibility count      =    12
              Mean for random partitions =    12.747
```

The insignificant BCPPTP indicates that the partitions are not significantly heterogeneous.

PARTPICA.EXE accepts up to 50 taxa and 1000 characters with up to five character states. It handles up to five partitions. An important limitation of this version of the program is that it ignores character interdependence as may be specified by integer codes. It also ignores the‘*’ symbol in taxon names and any constraints in the data file.

Programs for Evaluating Individual Characters

DNALQP.EXE calculates Le Quesne probabilities of individual characters by random permutation of those characters. For each character the program gives the number of observed incompatibilities, the number expected under the null hypothesis (the average for the random permutations), the Le Quesne probability, and the minimum number of incompatibilities observed from the random permutations with the corresponding minimum Le Quesne probability that could have been achieved by the character (as evidenced by the observations from random data).

An example output file from analysis of the data in BOX 1 is shown opposite.

Settings are mostly as for the MATRIX.EXE analysis shown above.

Taxon 1 (the hypothetical ancestor) was not included in the random permutation (but provided information on polarity).

Note that, in this example, the minimum Le Quesne probabilities are quite high. This reflects the small size of the data. There are only seven taxa, one of which was held constant, and thus few distinct permutations of characters are possible. If the Le Quesne probability were to be used as a test statistic with such few trials, what is significant should be considered carefully.

```
PICA 4.0 - 17/1/2001 16:18
DNALQP.EXE  - Le Quesne Probability randomisation test

*** ANALYSIS ***

example.dat 7 taxa 19 characters
 Duellman and Trueb (1986) caecilian data

*** SETTINGS ***

Approximation by 99 random character permutations
Character interdependence as specified by integer codes
Taxa held constant: 1
Missing entries held constant
Le Quesne probabilities => 0.0000

*** RESULTS ***

       Incompatibilities   Le Quesne        Minimum
  N     Obs    Expected   Probability   Probability/Obs
-----------------------------------------------------------
   1 :   2       12.07       0.09          0.09      2
   2 :   1       10.77       0.17          0.17      1
   3 :   3       12.44       0.06          0.06      3
   4 :  Equivalent to Character 2
   5 :  Equivalent to Character 1
   6 :  Equivalent to Character 3
   7 :   0        8.49       0.07          0.07      0
   8 :  Equivalent to Character 7
   9 :  15       12.02       0.70          0.05      1
  10 :   8       13.91       0.16          0.04      2
  11 :   5        8.68       0.37          0.06      0
  12 :  Equivalent to Character 1
  13 :   3       12.70       0.12          0.12      3
  14 :  Equivalent to Character 1
  15 :  Equivalent to Character 2
-----------------------------------------------------------
```

In this example, several of the characters that have ‘non-significant’ Le Quesne probabilities achieve the estimated minimum possible observed incompatibilities and therefore, judged by their compatibility they are as good as they can be given the structure of the data. The worst characters appear to be 9, 10 and 11 which have Le Quesne probabilities > 0.2.

Where characters are equivalent (identify the same split and have the same, if any, patterns of interdependence), their Le Quesne probabilities are identical. Thus there is no advantage to analysing sets of equivalent characters and DNALQP.EXE reports the results for the first character and the equivalence of the others. Characters that are parsimony uninformative, and for which the test is meaningless, are reported as such. In the example output characters 16-19 have been excluded, they are equivalent to character 2.

DNALQP.EXE accepts up to 50 taxa and 1000 characters with up to five character states. It accepts character interdependence as specified by integer codes, allows missing data and individual taxa to be held constant or randomised, allows a subset of characters to be tested and results to be output only for characters with Le Quesne probabilities higher than a user–specified threshold. It ignores data partitions or constraints in the data file. Optionally, it will output a data file including only characters with Le Quesne probabilities smaller than a user-specified threshold.

Programs for Evaluating Individual Characters

LQPROB.EXE calculates Le Quesne probabilities for individual binary characters by either random permutation of those characters, or exhaustive enumaration of all permutations, depending on whether the number of possible permutations is less than the number of random permutations selected by the user or not. It calculates the probabilities exactly when this requires less computation than estimating them through random permuatation. For each character the program gives the number of observed incompatibilities, the number expected under the null hypothesis (the average for the random permutations), the Le Quesne probability, and the minimum number of incompatibilities observed from the random permutations with the corresponding minimum Le Quesne probability that could have been achieved by the character, and the number of trials.

An example output file from analysis of the data in BOX 1 is shown opposite.

Settings are mostly as for the MATRIX.EXE analysis shown above.

Taxon 1 (the hypothetical ancestor) was not included in the random permutation but was used to specify polarities.

The results are largely in agreement with those from the DNALQP analyses. In all cases the number of possible permutations of characters was less than the number of random permutations specified so that exact Le Quesne probabilities were determined.

```
PICA 4.0 - 17/1/2001 18:14
LQPROB   - Le Quesne Probabilities for binary data

*** ANALYSIS ***

example.dat 6 taxa 19 characters
 2 0 1 Duellman and Trueb (1986) caecilian data

*** SETTINGS ***

Exact calculation by exhaustive enumeration
Approximation by 99 random character permutations
Polarity determined by hypothetical ancestor
Character interdependence as specified by integer codes

*** RESULTS ***

      Incompatibilities   Le Quesne       Minimum
 N    Obs      Expected  probability  probability/obs  trials
--------------------------------------------------------------
  1:    2      12.05     0.05           0.05    2       20
  2:    1      10.33      0.2            0.2    1        6
  3:    3      12.27     0.07           0.07    3       15
  4:  Equivalent to Character 2
  5:  Equivalent to Character 1
  6:  Equivalent to Character 3
  7:    0       8.40     0.07           0.07    0       15
  8:  Equivalent to Character 7
  9:   15      12.05     0.65           0.05    1       20
 10:    7      12.93     0.20           0.07    2       15
 11:    3       8.40     0.13           0.07    0       15
 12:  Equivalent to Character 1
 13:    2      11.70      0.1            0.1    2       10
 14:  Equivalent to Character 1
--------------------------------------------------------------
```

LQPROB.EXE accepts up to 100 taxa and 500 characters. Characters must be binary and must be coded using ‘0’s, and ‘1’s and with ‘?’s for missing entries. Characters may be polar or non-polar. Polarity is assumed when the first taxon in the matrix includes the ‘*’ symbol in its name, in which case its character states are considered primitive and any characters coded as ‘?’ are treated as non-polar. Otherwise all characters are treated as non-polar. It accepts character interdependence as specified by integer codes. Missing data are held constant and cannot be included in the randomisation in LQPROB.EXE. It allows a subset of characters to be tested and ignores data partitions or constraints in the data file. If you abort the program, results for characters that had been analysed to that point are written to file.

If you encounter problems with LQPROB.EXE try using DNALQP.EXE which is simpler (and therefore less likely to go wrong).

Programs for Evaluating Individual Characters

BOILDOWN.BAT controls the operation of SETPICA.EXE and BOILPICA.EXE that jointly implement boildowns and character ranking for binary data using CCSRs or Normal Deviates. The output includes matrix and character CCSRs and normal deviates and as observed and expected number of incompatibilities of each character. In each iteration of the boildown the worst characters are identified and deleted, and revised matrix CCSRs and normal deviates are given. Final values of these measures, and the composition of the core set are given and absolute weights are reported for all characters.

An example output file from an analysis of the data in BOX 1is shown opposite. Some results are excluded but the salient features remain.

This analysis used the CCSR to rank characters.

Settings are mostly as for the MATRIX.EXE analysis shown above.

Characters were treated as polar based on the coding of the hypothetical ancestor and a constraint used to prevent two characters from being deleted.

The matrix CCSR and Normal Deviate indicate that the data as a whole is far from random. Note that the values for the data change with the elimination of characters in the boildown giving a measure of increases or decreases in the apparent quality of the remaining data..

The boildown eliminates characters 9, 10m and 11 in that order, leaving a core set of 16 characters.

```
PICA 4.0 - 17/1/2001 20:42
BOILDOWN - Iterative character ranking using the
           Coefficient of Character State Randomness
*** ANALYSIS ***
7 taxa 19 characters
  2 0 1 Duellman and Trueb (1986) caecilian data
*** SETTINGS ***
Interdependence of characters recognised
Characters are polar
Constraint of 2 characters : 1 7
*** RESULTS ***
Matrix CCSR =   0.2270 Matrix Normal Deviate =  12.9117
     Incompatibililty
  N  Observed Expected     CCSR        Ndev
-----------------------------------------------
  1      2     11.9000     0.1681     4.6809
  2      1     10.4333     0.0958     4.4499
  4      1     10.6000     0.0943     4.5554
  5      2     11.4000     0.1754     4.5927
  6      3     13.0333     0.2302     5.0271
  7      0      8.6000     0.0000     3.8222
  9     15     12.2000     1.2295    -1.1600
 10      7     12.7333     0.5497     2.7113
 11      3      8.6000     0.3488     2.4065
 12      2     11.9000     0.1681     4.6809
 13      2     11.7000     0.1709     4.5463
 14      2     11.9000     0.1681     4.6809
 15      1     11.1000     0.0901     4.6539
-----------------------------------------------------------
       Averages for Informative Characters
 Obs =  2.5263 CCSR = 0.2154  Normal Deviate =  4.0482
-----------------------------------------------------------
*** BOILDOWN ***
1 - Deleted Character(s): 9  CCSR = 1.2295
 Reduced Matrix Normal Deviate = 14.0847 CCSR = 0.0962
 Absolute Character Weight = 1
2 - Deleted Character(s): 10  CCSR = 0.5028
 Reduced Matrix Normal Deviate = 13.8259 CCSR = 0.0368
 Absolute Character Weight = 3
3 - Deleted Character(s): 11  CCSR = 0.4018
 Reduced Matrix Normal Deviate = 14.0262 CCSR = 0.0000
 Absolute Character Weight = 6

Core set of 16 Compatible Characters
1 2 3 4 5 6 7 8 12 13 14 15 16 17 18 19
 Average of Original CCSRs in Core Set = 0.1227
 Absolute Character Weight of Core Set Characters = 12
```

BOILDOWN.BAT accepts up to 100 taxa and 500 characters. Characters must be binary and must be coded using ‘0’s, and ‘1’s and with ‘?’s for missing entries. Characters may be polar or non-polar. Polarity is assumed when the first taxon in the matrix includes the ‘*’ symbol in its name, in which case its character states are considered primitive and any characters coded as ‘?’ are treated as non-polar. Otherwise all characters are treated as non-polar. It accepts character interdependence and character constraints as specified in the data file. It ignores partitions.

SETPICA.EXE writes a temporary file during its execution. This can be quite large and it is recommended that the routine be run in a directory/folder on a hard drive, and not on a floppy disc.

Programs for Evaluating Splits

SPLIT.EXE calculates a priori SC and SSPTPs for given splits by random permutation of the character data. For each prespecified split it also reports its index, its conflict and its weighted support, and average conflict and weighted support values for randomly permuted data.

An example output file from an analysis of the data in BOX 1and the splits in BOX 2 is shown opposite.

Settings are mostly as for the MATRIX.EXE analysis shown above.

If the split is partial, the index is given as zero.

```
PICA 4.0 - 17/1/2000 17:37
SPLIT    - A priori split support and conflict randomisation tests

*** ANALYSIS ***

example.dat 7 taxa 19 characters example.spl 3 splits
 Duellman and Trueb (1986) caecilian data   caecilian splits

*** SETTINGS ***

SS- and SCPTPs approximated by 99 random data permutations
Character interdependence as specified by integer codes
Taxa held constant: 1
Missing entries held constant

*** RESULTS ***

-------------------------------------------------------------------
SPLIT      Index Conflict   PTP    Mean   Support     PTP    Mean
-------------------------------------------------------------------
0001111       15    3     0.01    13.95   2.5000     0.01   0.1465
00?1111        0    1     0.01    10.46   9.5000     0.01   1.4899
0101010       42   19     1.00    13.17   0.0000     1.00   0.3131
-------------------------------------------------------------------
```

The results indicate that the first split shows far less conflict with the original data than it does with randomly permuted data. Similarly, the split is far better supported by the original data than by randomly permuted data. The similar partial split is also well supported. Note that the splits differ only in the exclusion of the Ichthyophiidae (taxon 3) from the latter, and that this yields greater support and lower incompatibilities/conflict. Changes in the reverse direction with the elimination of taxa are not possible. The last split is essentially random with respect to background knowledge of caecilian phylogeny and has never been entertained. The analysis shows that this split is no better supported by the real data than by random data and that it has it has significantly more conflict with the real data than it has with randomly permuted data. The latter suggests that the data contain significant evidence of the non-monophyly of either group identified by the split.

SPLIT.EXE accepts up to 50 taxa and 1000 characters, with up to five character states. It accepts character interdependence as specified by integer codes, allows missing data and individual taxa to be held constant or randomised, and accepts full or partial splits. It ignores partitions and constraints in the data file.

Programs for Evaluating Splits

ALROY.EXE calculates a priori SNPTPs for given splits by random permutation of polar binary character data. For each prespecified split it also reports the split, its index, its nesting count, and the average nesting count for for randomly permuted data.

An example output file from an analysis of the data in BOX 1and the three splits in BOX 2 is shown opposite.

Settings are mostly as for the MCPTP.EXE analysis shown above.

The results are parallel those of the split support test reported by the SPLIT program (above).

```
PICA 4.0 - 20/3/2000 21:20
ALROY    - A priori split nesting randomisation tests

*** ANALYSIS ***

example.dat 7 taxa 19 characters example.spl 3 splits
 2 0 1 Duellman and Trueb (1986) caecilian data
caecilian splits

*** SETTINGS ***

SNPTPs approximated by 99 random data permutations
Character interdependence as specified by integer codes
Taxa held constant: 1
Missing entries held constant

*** RESULTS ***

-------------------------------------------------
SPLIT      Index    Nesting       PTP     Mean
-------------------------------------------------
0001111       15       16        0.01    4.626
00?1111        0       18        0.01    7.970
0101010       42        0        1.00    4.980
-------------------------------------------------
```

ALROY.EXE accepts up to 50 taxa and 1000 characters. Characters must be binary, coded with ‘0’ and ‘1’, and are assumed to be polar with ‘1’ derived. It accepts character interdependence as specified by integer codes, allows missing data and individual taxa to be held constant or randomised, and accepts full or partial splits. It ignores partitions and constraints in the data file.

Programs for Evaluating Splits

RPTP.EXE calculates SC and SSRPTPs for given splits by random permutation of these splits. For each prespecified split it also reports the split, its index, its conflict and its weighted support, and average conflict and weighted support values for randomly permuted data. Comparable random splits with which the original split is compared are splits of the same size as the original. Optionally, the comparison can also be restricted to only those splits that are incompatible with the original split.

An example output file from an analysis of the data in BOX 1and the splits in BOX 2 is shown opposite.

Settings are mostly as for the MCPTP.EXE analysis shown above.

Results are similar to the a priori split PTP tests shown above.

```
PICA 4.0 - 19/3/2000 17:41
RPTP     - Split support and conflict R-randomisation tests

*** ANALYSIS ***

example.dat 7 taxa 19 characters; example.spl 3 splits
 Duellman and Trueb (1986) caecilian data   caecilian splits

*** SETTINGS ***

SS- and SCRPTPs approximated by 99 random split permutations
All comparable splits used
Missing entries of Partial Splits held constant

*** RESULTS ***

------------------------------------------------------------------
SPLIT      Index Conflict  RPTP    Mean   Support     RPTP   Mean
------------------------------------------------------------------
0001111       15     3     0.03    14.23   2.5000     0.03  0.1263
00?1111        0     1     0.10    10.46   9.5000     0.07  0.6364
0101010       42    19     1.00    12.91   0.0000     1.00  0.2626
------------------------------------------------------------------
```

Note however, that the partial split has higher RPTPs than PTPs. Note also that (as in the Le Quesne probability tests) the number of trials is low, and ‘significance’ must be considered carefully. For example, there are only 15 permutations of the partial split.

RPTP.EXE works for up to 50 taxa and 1000 characters, with up to five character states. It accepts full and partial splits, allows missing taxa in partial splits to be held constant or randomised, and makes comparisons with either any equivalent splits or only those that are incompatible with the original; split. It ignores integer codes, partitions and constraints in the data file amnd the ‘*’ symbol in taxon names.

Programs for Evaluating Splits

FAITH.BAT controls the operation of four programs (FAITH1-4.EXE) that jointly determine a posteriori SS- and SCPTPs for all full splits supported by a set of data, by random permutation of the data. For each split FAITH.BAT gives the split index, the split itself, its size, its weighted support, the corresponding SSPTP, its conflict, and the corresponding SCPTP. Determination of a posteriori SS- and SCPTPs is achieved by creating a number of random data sets, identifying the full splits supported by the original and random data sets, their size and their levels of support and conflict. The scores for each split in the original data are compared to the best scores for same sized splits in the random data. Optionally, all characters, only those that are parsimony informative or only those that are split informative can be used.

```
PICA 4.0 - 19/1/2001 17:54
FAITH    - A posteriori split support and conflict
randomisation tests

*** ANALYSIS ***

7 taxa 19 characters
 Duellman and Trueb (1986) caecilian data

*** SETTINGS ***

SS- and SCPTPs approximated by 99 random data permutations
Missing entries held constant
Using all characters

*** RESULTS ***

--------------------------------------------------------------
    Index SPLIT  Size  Support      PTP  Conflict  PTP
--------------------------------------------------------------
        7 0000111 3     4.0000     0.02     2     0.01
       31 0011111 2     7.0000     0.01     1     0.01
       15 0001111 3     2.5000     0.07     3     0.01
        3 0000011 2     2.0000     0.92     0     0.01
       35 0100011 3     1.0000     1.00    15     1.00
       27 0011011 3     1.0000     1.00     7     0.03
       24 0011000 2     1.0000     1.00     3     0.03
*** END ***
```

An example output file from analysis of the data in BOX 1 is shown opposite.

Settings are mostly as for the MATRIX.EXE analysis shown above.

The seven splits constitute a spectrum of the support in the data.

Three of these have minimal support and do not pass the split support test. Note however that two of them do pass the more liberal conflict test. The third has significantly more conflict with the data than do comparable splits with random data –strong evidence for non-monophyly. Only the three best supported groups pass the support test.

FAITH.BAT works for up to 100 taxa and 5000 characters, with up to five character states. It will accept up to 20,000 characters provided that the number of split informative or parsimony informative characters does not exceed 5,000 and the option selecting this set of characters is chosen. It reports results only for full splits supported by the data, and allows missing data to be held constant or randomised. It ignores integer codes, partitions and constraints in the data file, and the ‘*’ symbol in taxon names.

Index numbers of splits our output if the number of taxa is less than 49. If the number of taxa is higher the split is given but not its index.

A number of temporary files are written during the execution of this computationally demanding routine. These can be quite large and it is recommended that the routine be run in a directory/folder on a hard drive, and not on a floppy disc.

Programs for Evaluating Taxa

JACTAX.EXE calculates the incompatibilities caused by particular terminal taxa by first order taxon jackknife. It optionally reports average scores for randomly generated taxa and reports a corresponding tail probability describing the proportion of taxa (real and random) causing as little conflict as the real taxon. Alternatively it reports the conflict caused by taxa determined by a second order taxon jackknife with no simulation using randomly generated data. Conflicts are reported as absolutes and as percentages of the total conflict in the data. Missing entries can play an important part in the compatibility of taxa and the percentage of missing entries in the taxons character data is also reported. The program refers to taxa by numbers corresponding to their order of appearance in the data file, but outputs a list of the taxon names corresponding to the numbers.

An example of output from two analysis of the data in BOX 1 is shown opposite. The list of taxon names has been ommitted.

Settings are mostly as for the MATRIX.EXE analysis shown above.

In the first analysis no randomisation was used and only the first and second order jackknife results are reported along with the percentage missing entries of each selected taxon.

In the second analysis each taxon was replaced with 99 randomly generated taxa and the mean number of comflicts caused by random taxa and a corresponding tail probabilities (P) for the proportion causing as few conflicts as the real taxon are reported..

```
PICA 4.0 - 29/1/2001 13:16
JACTAX   - Taxon Jackknife Compatibility Measures and Tests

*** ANALYSIS ***

example.dat 7 taxa 19 characters
 Duellman and Trueb (1986) caecilian data

*** SETTINGS ***

First and second order jackknife - No randomisation
Character interdependence as specified by integer codes
Taxa excluded: 1

*** RESULTS ***

Matrix Incompatibility Count = 24

------------------------------------------
              Incompatibilities
           1st Order       2nd Order
Taxon  %?  Caused  %       Caused   %
------------------------------------------
  2   0.00   15  62.500    16.143 67.262
  3   0.00    5  20.833     9.857 41.071
  4   0.00    4  16.667     9.143 38.095
  5   5.26   10  41.667    12.571 52.381
  6   0.00    0   0.000     8.714 36.310
  7   0.00    0   0.000     8.714 36.310
--------------------------------------------
             Incompatibilities
Taxon  %?  Caused  %        Mean       P
--------------------------------------------
  2   0.00   15  62.500    31.566     0.08
  3   0.00    5  20.833    24.000     0.04
  4   0.00    4  16.667    24.525     0.04
  5   5.26   10  41.667    22.768     0.08
  6   0.00    0   0.000    29.444     0.01
  7   0.00    0   0.000    27.556     0.01
--------------------------------------------
```

Tail probabilities from the comparisons with randomly generated data may need to be interpreted carefully. In the example two taxa (6 and 7) cause no conflict. This is because they do not differ in character states. ‘Significant’ tail probabilities may indicate no more than the presence of similar taxa in the data. The results do however, indicate that taxa 1 and 5 are particularly responsible for the conflict in the data. Although the tail probabilities for these taxa are not significantly different from random, this does not necessarily indicate that the taxa should be eliminated from any analyses. Eliminating taxa from analyses carries its own dangers (Wilkinson, 1995b). The results may however prompt a re-evaluation of their morphology or the attempt to improve their performance by identifying and possibly eliminating or downweighting the problematic characters.

JACTAX.EXE works for up to 50 taxa and 1000 characters, with up to five character states. It provides options for randomly generating data or for performing a second order taxon jackknife. It accepts character interdependence as specified by integer codes so that randomly generated taxa cannot cause conflict between interdependent characters that cannot be incompatible. Missing entries are not replaced by character datat in randomly generated taxa. Individual taxa can be excluded from tests, but their data is included in the tests of other taxa. It ignores partitions and constraints in the data file.

RUNNING THE PROGRAMS

Most of the programs are stand alone entities. To run a program enter its name at the DOS prompt (or click on its name/icon under Windows). Performing a boildown or calculating a posteriori SS- and SCPTPs requires several programs to be run sequentially. This can be done using the DOS batch files BOILDOWN.BAT and FAITH.BAT respectively. As with the stand alone programs enter the batch file name or click on its icon.

User input: Most of the programs require some input from the keyboard. The required input may include filenames, single letters, numbers, or carriage returns. Be careful when entering data from the keyboard: not all the programs have full error error-checking. When programs ask a question requiring a yes/no type answer you must use the single letter '**y**' or '**n**' [**N.B. you must use lower case**]

Filenames: - Most programs call for the names of input, output and where applicable split files. If specified input data or split files do not exist you will be prompted for another file name. If output files already exist you will be asked if you want to overwrite them.

Characters: Some programs allow you to focus on a subset of characters. Where the programs require the characters to be specified from the keyboard they must be referred to by the number of the column they occupy in the character data (between one and the total number of characters). In such cases the programs can only deal with one contiguous block of characters and will prompt for the numbers of the first and last characters in the block.

Aborting operation: You can abort the execution of most of the programs by hitting CONTROL-BREAK, although you may sometimes have to wait a little while (as the program exits a subroutine) for this to take effect. Where the programs detect serious errors they may prompt you to take this action. With the Le Quesne probability, boildown, and split rPTP programs results up to the point where the run is aborted will have been written to the output file.

Specific questions and responses: Some of the potential problems are that may be encountered in running the programs will produce specific error messages which should be self explanatory. Most of the other questions or instructions to which you must respond are described here.

```
Select a contiguous subset of characters to test?
```

Runs of the Le Quesne probability programs LQPROB.EXE and DNALQP.EXE may be very time consuming. Results are written to output as they are accumulated so that aborting the program will not cause the loss of all results. This option allows you to specify that tests are to be performed for a particular subset of the characters/sites. Ignored characters are used in the calculation of compatibility scores for included characters but not themselves tested.

```
Include results for characters with Le Quesne probability =>
```

Responses to this prompt control the output of DNALQP.EXE. The prompt requires a number between 0 and 1. Entering '0' (zero) ensures that results for all characters are written to the output file. Higher values will exclude results for characters with Le Quesne probabilities lower than the specified value. Thus results will be for the worst characters.

```
Write characters passing the test to a new data set?
Include results for characters with Le Quesne probability <=
```

Responses to these prompts allow DNALQP.EXE to create a new data file including only those characters that have Le Quesne probabilities lower than the user-specified value (which must be a number between 0 and 1).

```
Maintain original positions of missing entries?
```

This allows you to either include missing entries in the random permutation or to maintain their positions as in the original data. A persistent bug affects some programs when missing data is maintained in its original positions and a selection of a subset of characters is also made.

```
Maintain positions of missing entries in partial splits?
```

This allows the test of partial splits in RPTP.EXE to focus upon partial splits including just those taxa included in the original partial split.

```
Enter number of trials (random permutations) :
```

This must be a positive integer. In LQPROB.EXE, if this number exceeds the number of permutations of the binary character being tested then the Le Quesne probability will be determined by exhaustive enumeration. Otherwise tests are by random permutation. The exhaustive method not only provides a more precise measure, but it may also reduce run times.

```
Partition detected – Select Treatment
     0 – Ignore
     1 – Randomly permute both partitions
     2 – Hold Partition 1 constant
     3 – Hold Partition 2 constant
Enter your selection:
```

The MATRIX.EXE and NATRIX.EXE programs will detect partitions if they are included in the data file. Options governing the treatment of partitions are specified and are selected by entering the corresponding number. The detected partitions can be ignored. Alternatively, if one or both can be randomly permuted and results will include WCPTPs for the randomised partitions and BCPTPs for the between partition incompatibility. Options 2 or 3 provide a means of restricting the random permutation to a subset of characters and thereby performing a conditional test (see Wilkinson, 1997a for an example).

```
Symbols statement expected.
Press enter to use default settings; Control-break to abort.
```

If no symbols statement is found by any of the programs for multistate data, you will be asked whether the program should continue using the default symbol set. The case sensitive default is for DNA data using the symbols 'A', 'C', 'G', 'T', and '-', with gaps ('-') treated as a fifth character state. If you do not wish to treat alignment gaps as a fifth state exclude the '-' symbol from your symbols statement, and, along with all other unspecified symbols, it will be treated as representing missing entries.

```
Compare only with incompatible comparable splits?
```

The RPTP.EXE program allows specified splits to be compared with any comparable (i.e. same size) splits or with only those comparable splits that are incompatible with the original. The latter would seem to be a reasonable restriction if the behaviour of compatible equivalent splits is considered irrelevant to the evaluation of the original split (Wilkinson, 1998a).

```
Use: 1 - All data
     2 - Split informative data
     3 - Parsimony informative data
Enter your selection:
```

The first part of the FAITH routine (FAITH1.EXE) allows the user to restrict attention to subsets of the data. Options are specified and are selected by entering the corresponding number. Analytically, using all the data is the same as using only split informative data (uninformative data is ignored) but the latter is more efficient and may allow larger data sets (more characters) to be analysed. The third option allows only parsimony informative characters to be used in the tests should this be desired.

```
Choose ranking criterion : 1. CCSR
                           2. Normal deviate
```

BOILPICA.EXE offers the choice of ranking characters by coefficients of character state randomness or normal deviates. To run both analyses you will need to run the program twice.

```
Select model of random taxon generation
     0 - No Randomisation
     1 - Character States Equiprobable
     2 - Use Empirical Frequencies
Enter your selection :
```

JACTAX.EXE offers three options. The first option results in first and second order taxon jackknifing with no randomisation. The seconmd and third options result in first order taxon jackknifing and simulation tests using taxa generated randomly according to the selected model.

ACKNOWLEDGEMENTS

Initial development of PICA was supported by SERC Grant GR/F 87912 to Mike Benton, Dept. Geology, University of Bristol. Mike's support and that of SERC are gratefully acknowledged. I'm grateful to Dom Kreuzer for algorhythmics, to Fred Kraus for helping to identify problems with previous releases, and to Vince Smith, Charles Strong and Simon Harris for helping test some of the programs. It is a pleasure to acknowledge the intellectual debt I owe to those who have pioneered compatibility methods for stimulating my interest. especially Walter Le Quesne, George Estabrook, Buck McMorris, Chris Meacham, Garth Underwood, Mike Sharkey, to David Penny and his many colleagues for their work on splits, and to those who have pioneered the use of randomisation tests especially James Archie, Dan Faith, Peter Cranston and John Alroy.

REFERENCES

Alroy, J. 1994. Four permutation tests for the presence of phylogenetic structure. *Syst. Biol.* **43**: 430-437.

Archie, J. W. 1989a. A Randomisation test for phylogenetic information in systematic data. *Syst. Zool.* **38**: 239-252.

Archie, J. W. 1989b. Homoplasy excess ratios: new indices for measuring levels of homoplasy in phylogenetic systematics and a critique of the consistency index. *Syst. Zool.* **38**: 253-269.

Bandelt, H.-J. & Dress, A. 1992. A canonical decomposition theory for metrics on a finite set. *Adv. Math.* **92**: 47-105.

Baum, B. R. 1992. Combining trees as a way of combining data sets for phylogenetic inference, and the desirability of combining gene trees. *Taxon* **41**: 3-10.

Day, W. H. E., Estabrook, G. F. & McMorris, F. R. 1998. Measuring the phylogenetic randomness of biological data sets. *Syst. Biol.* **47**: 604-616.

Duellman, W. E. & Trueb, L. 1986. *Biology of Amphibians*. McGraw-Hill Book Co., New York.

Estabrook, G. F. & Landrum, L. 1975. A simple test for the simultaneous divergence of two amino acid positions. *Taxon* **24**: 609-613.

Estabrook, G. F., Strauch, J. G. & Fiala, K. L. 1977. An application of compatibility analysis to the Blackith's data on orthopteroid insects. *Syst. Zool*. **26**: 269-276.

Faith, D. P. 1991. Cladistic permutation tests for monophyly and nonmonophyly. *Syst. Zool.* **40**: 366-375.

Faith, D. P., & Cranston, P. S. 1991. Could a cladogram this short have arisen by chance alone?: On permutation tests for cladistic structure. *Cladistics* **7**: 1-28.

Farris, J. S. 1969. A successive approximations approach to character weighting. *Syst. Zool*. **18**: 374-385.

Farris, J. S. 1973. On comparing the shapes of phylogenetic trees. *Syst. Zool.* **22**: 50-54.

Farris, J. S., Källersjö, M. Kluge, A. C. & Bult, C. 1994. Testing significance of incongruence. *Cladistics* **10**: 315-319.

Felsenstein, J. 1978. Cases in which parsimony or compatibility methods will be positively misleading. *Syst. Zool.* **27**: 401-410.

Felsenstein, J. 1995. PHYLIP: Phylogenetic inference package. Version 3.5c. Computer programs and documentation. Department of Genetics, University of Washington, Seattle.

Fiala, K. 1984. CLINCH 6.2. State University of New York.

Foulds, L. R., Hendy, M. D. & Penny, D. 1979. Ageneral approach to proving the minimality of phylogenetic trees illustrated by an example with a set of 23 vertebrates. *J. Mol. Evol.* **13**: 151-166.

Frohlich, M. W. & Estabrook, G. F. 2000. Wilkinson support calculated with exact probabilities: an example using Floricaula/LEAFY amino acid sequences that compares three hypotheses involving gene gain/loss in seed plants. *Mol. Biol. Evol.* **17**: 1914-1925.

Gauld, I. & Underwood, G. 1986. Some applications of the Le Quesne compatibility test. *Biol. J. Linn. Soc*. **29**: 191-222.

Guise, A., Peacock, D. & Gleaves, T. A. 1982. A method for the identification of parallelism in discrete character sets. *Zool. J. Linn. Soc*. **74**:293-303.

Huelsenbeck, J. P., Bull, J. J. & Cunningham, C. W. 1996. Combining data in phylogenetic analysis. *TREE* **11**: 152-158.

Jakobsen, I. B. & Easteal, S. 1996. A program for calculating and displaying compatibility matrices as an aid in determining reticulate evolution in molecular sequences. *Comput. Appl. Biosci.* **12**: 291-295.

Kluge, A. G. 1976. Phylogenetic relationships in the lizard family Pygopodidae: an evaluation of theory, methods and data. *Misc. Pub. Mus. Zool. Univ. Michigan* **152**: 1-72.

Le Quesne, W. J. 1969. A method of selection of characters in numerical taxonomy. *Syst. Zool.* **18**: 201-205.

Le Quesne, W. J. 1972. Further studies based on the uniquely derived character concept. *Syst. Zool.* **21**: 281-288.

Le Quesne, W. J. 1989. The normal deviate test of phylogenetic value of a data matrix. *Syst. Zool*. **38**: 51-54.

Lento, G. M., Hickson, R. E., Chambers, G. K. & Penny, D. 1995. Use of spectral analysis to test hypotheses on the origin of pinnipeds. *Mol. Biol. Evol.* **12**: 28-52.

Manly, B. F. J. 1991. *Randomisation and Monte Carlo methods in Biology*. Chapman and Hall, London.

Meacham, C. A. 1981. A probability measure for character compatibility. *Math. Biosci.* **57**: 1-18.

Meacham, C. A. 1984. The role of the hypothesised direction of characters in the estimation of evolutionary history. *Taxon* **33**: 26-38.

Meacham, C. A. 1994. Phylogenetic relationships at the basal radiation of Angiosperms: further study by probability of character compatibility. *Syst. Bot*. **19**: 506-522.

Meacham, C. A. & Estabrook, G. F. 1985. Compatibility methods in systematics. *Ann. Rev. Ecol. Syst.* **16**: 431-446.

Mes, T. H. M. 1998. Character compatibility of molecular markers to distinguish asexual and sexual reproduction. *Mol. Ecol.* **7**: 1219-1727.

Page, R. D. M. 1989. Comments on component-compatibility in historical biogeography. *Cladistics* **5**: 167-182.

Penny, D., Watson, E. E., Hickson, R. E. & Lockhardt, 1993. Recent progess with methods for evolutionary trees. *NZ. J. Bot.* **31**: 275-288.

Purvis, A. 1995. A modification to Baum and Ragan's method for combining phylogenetic trees. *Syst. Biol.* **44**: 251-255.

Ragan, M. A. 1992. Phylogenetic inference based on matrix representation of trees. *Mol. Phy. Evol.* **1**: 53-58.

Rodrigo, A. G. 1992. Two optimality criteria for selecting subsets of most parsimonious trees. *Syst. Biol*. **41**: 33-40.

Rodrigo, A. G. 1996. On combining cladograms. *Taxon* **45**: 267-274.

Salisbury, B. A. 1999. Strongest evidence in compatibility: clique and tree evaluation using apparent phylogenetic signal. *Taxon* **48**: 755-766.

Sharkey, M. J. 1989. A hypothesis-independent method of character weighting for cladistic analysis. *Cladistics* **5**: 63-86.

Sharkey, M. J. 1994. Discriminate compatibility measures and the reduction routine. *Syst. Biol.* **43**: 526-542.

van der Hulst, R. G. M., Mes, T. H. M., den Nijs, J. C. M. & Bachmann, K. 2000. Amplified fragment length polymorphism (AFLP) markers reveal that population structure of triploid dandelions (*Taraxacum officinale)* exhibits both clonality and recombination. *Mol. Ecol.* **9**: 1-8.

Wilkinson, M. 1992a. Consensus, compatibility and missing data in phylogenetic inference. Unpubl. PhD thesis. University of Bristol.

Wilkinson, M. 1992b. PERMUTE software and documentation. Department of Geology, University of Bristol.

Wilkinson, M. 1994a. Weights and ranks in numerical phylogenetics *Cladistics* **10**: 321-329.

Wilkinson, M. 1994b. Common cladistic information and its consensus representation: reduced Adams and reduced cladistic consensus trees and profiles. *Syst. Biol.* **43**: 343-368.

Wilkinson, M. 1995a. More on reduced consensus methods. *Syst. Biol.* **44**: 435-439.

Wilkinson, M. 1995b. Coping with missing entries in phylogenetic inference using parsimony. *Syst. Biol*. **44**:501-514.

Wilkinson, M. 1995c. PICA95. Software and documentation. School of Biological Sciences. University of Bristol.

Wilkinson, M. 1996. Majority-rule reduced consensus methods and their use in bootstrapping. *Mol. Biol. Evol.* **13**: 437-444.

Wilkinson, M. 1997a. Characters, congruence and quality: a study of neuroanatomical and traditional data in caecilian phylogeny. *Biological Reviews* **72**: 423-470.

Wilkinson, M. 1997b. Limitations on the use of compatibility methods for polarizing and ordering characters. *Syst. Biol.*, **46**: 230-231.

Wilkinson, M. 1997c. On phylogenetic relationships within *Dendrotriton* (Amphibia: Caudata: Plethodontidae): is there sufficient evidence? *Herpetol. Journal* **7**: 55-65.

Wilkinson, M. 1998a. Split support and split conflict randomisation tests in phylogenetic inference. *Syst. Biol.* **47**: 673-695.

Wilkinson, M. 1998b. Higher-order homoplasy tests. *Acta Biotheoretica* **46**: 109-116.

Wilkinson, M. & Nussbaum, R. A. 1996. On the phylogenetic position of the Uraeotyphlidae (Amphibia: Gymnophiona). *Copeia* **1996**: 550-562.

Appendix - Summary of randomisation tests in PICA

Full name	Acronym	Synonyms	Program
Matrix Conflict	MCPTP	Compatibility count (CC) PTP (Wilkinson, 1992a,b) PC – permutation compatibility test (Alroy, 1994) Pairwise compatibility (PC) PTP (Wilkinson, 1995c, 1997a,c) C_f (Day et al.,1998).	MATRIX.EXE
Matrix Nesting	MNPTP	PH – permutation hierarchy test (Alroy, 1994) Hierarchical pairs (HP)PTP (Wilkinson, 1995c)	NATRIX.EXE
Between Partition Conflict	BCPTP	Pairwise compatibility (PC) PTP (Wilkinson, 1995c, 1997a)	MATRIX.EXE
Between Partition Nesting	BCPTP	-	NATRIX.EXE
Between Partition Conflict Partition	BCPPTP	Pairwise compatibility (PC) PTP (Wilkinson, 1995c, 1997a)	PARTPICA.EXE
Within Partition Conflict	WCPTP	Pairwise compatibility (PC) PTP (Wilkinson, 1995c, 1997a)	MATRIX.EXE
Within Partition Nesting	WNPTP	-	NATRIX.EXE
Within Partition Conflict Partition	WCPPTP	Pairwise compatibility (PC) PTP (Wilkinson, 1995c, 1997a)	PARTPICA.EXE
Le Quesne Probability	LQP	Frequency of Compatibility Attainment (Meacham, 1994)	DNALQP.EXE LQPROB.EXE
A priori Split Conflict	SCPTP	NPC – nodal permutation compatibility test (Alroy, 1994)	SPLIT.EXE
A priori Split Nesting	SNPTP	NPH – nodal permutation hierarchy test (Alroy, 1994)	ALROY.EXE
A priori Split Support	SSPTP	-	SPLIT.EXE
A posteriori Split Conflict	SCPTP	-	FAITH.BAT
A posteriori Split Support	SSPTP	-	FAITH.BAT
Split Conflict Relationship	SCRPTP	Equivalent split conflict (Wilkinson, 1995c, 1998a)	RPTP.EXE
Split Support Relationship	SSRPTP	Equivalent split support (Wilkinson, 1995c, 1998a)	RPTP.EXE