Principles of interactive keys

1 January 2013

M.J. Dallwitz, T.A. Paine and E.J. Zurcher

This is an updated and expanded version of a paper presented at a conference on 'Computer-based Species Information', held at the University of Kent at Canterbury, UK, in December 1996 (Dallwitz, Paine and Zurcher 1998).

Contents

Abstract

Interactive keys can offer several advantages over conventional keys:

- Any characters can be used, and their values changed, in any order.
- A correct identification can be made in spite of errors by the user or in the data.
- Numeric characters can be used directly, without being divided into ranges.
- The user can express uncertainty by entering more than one state value, or a range of numerical values.

Other desirable features include:

- Advice on the most suitable characters to use at any stage of an identification.
- Locating errors that were circumvented by the error-tolerance mechanism.
- Provision for gaps in the values recorded for integer numeric characters.
- Character dependencies: certain character values making other characters inapplicable.
- Storing, searching, and displaying free-text information.
- Use of probabilities.
- Provision for restricting any operation to subsets of the characters and taxa.
- Glossaries and notes on interpretation of characters.
- Provision for information retrieval.
- Finding the differences and similarities between taxa.
- Finding diagnostic descriptions.

- Illustrations of characters and taxa.
- The ability to handle large data sets efficiently.
- Data sharing with other description-based applications: description writing, generation of conventional keys, and phenetic and cladistic analysis.

It is an inevitable consequence of the flexibility of interactive keys that much of the strategy involved in carrying out an identification is left to the user. Good strategies must be learnt if the keys are to be used to the best advantage.

Introduction

Identification is the process of finding the taxon to which a specimen belongs. Several methods are available for aiding this process (e.g. Pankhurst 1991). The most important are conventional identification keys and interactive keys.

A conventional identification key is a tree with characters at the internal nodes and taxon names at the terminal nodes. Each branch corresponds to a state of the character or characters at the node from which it arises. The user starts at the root of the tree, and follows the branches corresponding to the character states exhibited by the specimen until the taxon name is reached.

Authors of conventional keys try to provide some flexibility for the user by placing alternative characters at each node, but the possibilities for doing this are limited, because the characters must have identical distributions of their states among the taxa remaining in contention at that node. An error by the user in assigning a character state to the specimen inevitably leads to a wrong identification, unless the author has allowed for the possibility of this error by placing the taxon name in the subtree corresponding to the wrongly assigned state, as well as in the subtrees corresponding to states actually exhibited by the taxon. The author's use of this mechanism must also be limited, because each possible error (taxon/character-state combination) treated in this way adds a terminal node to the tree. This increases the size of the printed key (proportional to the number of terminal nodes), and the average number of characters that must be used to obtain an identification (proportional to the logarithm of the number of terminal nodes).

After any identification, it is good practice to check its accuracy by comparing the specimen with a description or illustrations of the taxon, or with other specimens known to belong to the taxon. When a conventional key is being used, the only way to recover from a wrong identification due to an error by the user is to guess where the error was made, return to that node, and try following another branch. If the error is in the key itself (that is, an error was made by the author), recovery is not possible.

An interactive key is an interactive computer program in which the user enters attributes (character-state values) of the specimen. Typically, the program eliminates taxa whose attributes do not match those of the specimen, and the process is continued until only one taxon remains. (However, some programs just sort the taxa on the degree of matching to the specimen.) The taxon attributes are usually stored as a characters-by-taxa 'matrix'. It is also possible to store the attributes as 'rules', but this kind of program is generally less satisfactory (Dallwitz 1992).

The defining characteristic of interactive keys is that characters can be used in any order. Conventional keys implemented on computers, for example, by using hypertext links to define the paths through the keys, are not interactive keys. (However, they may offer some advantages over printed keys, for example, easier access to character illustrations and notes.)

The criteria presented here are based on experience with the interactive identification program Intkey (Watson et al. 1989; Dallwitz 1993; Dallwitz, Paine and Zurcher 1993a; Dallwitz, Paine and Zurcher 1995), which uses data prepared in DELTA format (Dallwitz 1980). They were first published, in summary form, on the Taxacom mailing list (Dallwitz 1994), and have since been kept up to date (Dallwitz 1997).

Intkey was originally developed from an early version of the program Online (Pankhurst and Aitchison 1975). Many other interactive identification programs are available — see, for example, Edwards and Morse (1995), Dallwitz (1996).

All of the features mentioned below are available in Intkey, unless otherwise stated. For a detailed comparison of Intkey and some other interactive identification programs, see Dallwitz (2000).

Advantages over conventional keys

A well designed interactive key has several advantages over a conventional key.

Unrestricted character use

Any characters can be used, in any order (apart from restrictions imposed by 'Character dependencies' — see below). Characters which are not available on the specimen, or whose interpretation is not clear to the user, can be avoided (provided that there is sufficient redundancy in the data — see 'Construction of interactive keys' below).

Character deletion and changing

The values of any character can be changed at any stage of the identification, or any character deleted from the identification. An undesirable limitation is to allow only the changing or deletion of the last character used.

Error tolerance

A correct identification can be made in spite of errors by the user or in the data. Taxa are normally eliminated when they differ from the specimen in any way. If it is known or suspected that an error has been made, the program can be instructed to eliminate taxa only if they differ from the specimen in more than one attribute. It is immaterial where the error occurred, and whether it was made by the user or by the author of the data.

In Intkey, this function is controlled by the 'Tolerance' parameter, whose value may be 0 or any positive integer. Taxa are eliminated if they differ from the specimen in more attributes than the current value of 'Tolerance'. The parameter may be set to any permitted value at any time in the identification process, but typically it would be incremented by 1 when an identification has been made and found to be incorrect. The identification process is then continued, exactly as before. If *all* the taxa are eliminated, the program can increment 'Tolerance' automatically. If a single taxon remains, the program has no way of knowing whether this is the correct identification, and it is up to the user to check the identification, and, if necessary, increment 'Tolerance' manually.

Locating errors

The program should be able to locate user and/or data errors that were circumvented by the error-tolerance mechanism. The identification of user errors helps to improve the user's interpretation of characters. Data errors can be reported to the author for correction in later versions.

In Intkey, errors can be located by using the 'Differences' command to display the differences between the specimen and the remaining taxon.

Expressing uncertainty

The user can express uncertainty by entering more than one state value, or a range of numerical values. A user who is not sure which character-state value applies to the specimen may nevertheless sometimes be confident that some state values *do not* apply. Entering all the values that may conceivably apply to the specimen eliminates those taxa that never exhibit any of those values.

Numeric characters

Numeric characters can be used directly, without being divided into ranges. In conventional keys, numeric characters such as lengths must be divided into ranges before being incorporated in the key. This

usually results in loss of information. In an interactive key, the actual range of values exhibited by each taxon can be recorded in the data, and the taxon eliminated if the specimen's value does not fall within this range.

Easy updating

The key is maintained simply by making corrections and additions to the data matrix. Updating of conventional keys is relatively difficult. Even when the key is generated by computer from a data matrix, major changes to the matrix, particularly the addition of new characters and taxa, can have a large effect on the key structure, which has to be checked and possibly re-optimised.

Guidance in character selection

Best characters

The program should be able to advise the user on the most suitable characters for use at any stage of an identification. Because of the very large number of paths that may be taken through an interactive key, the ranking of the characters should be calculated directly from the data matrix for the set of taxa actually remaining at each stage of the identification. It is unsatisfactory to pre-assign rankings for a relatively small number of cases, as, for example, in a rule-based expert system.

The character-ranking algorithm used in Intkey is the same as that used in the key-generation program, Key (Dallwitz 1974). Unlike most such algorithms, it has a theoretical basis and gives sensible results for characters with three or more states, and for numeric characters. The relative weight of the separating power and the 'Character reliability' (see below) can be controlled by both the author and the user.

Ignoring character reliabilities, the best 2-state characters are those that divide the remaining taxa into groups that are as nearly equal as possible. Using such characters, the number of steps required to identify any taxon is about $\log_2 n$ (the formula is exact if the number of taxa is a power of 2). The worst characters, apart from those that do not differentiate the taxa at all or that differentiate only parts of taxa, separate only one taxon from the rest. Using such characters, the average number of steps required for an identification is $(1+2+\ldots n-1+n-1)/n$, i.e. $(n-1)(n+2)/(2n)$. For example, for 64 taxa the best key requires 6 steps, and the worst key requires an average of 32.5 steps.

Ranking of the characters can take a considerable time in large data sets, so it is important that the computation is as efficient as possible, and that the user does not have to wait for the ranking to be completed before choosing a character.

In Intkey, the characters found so far are displayed after 2 seconds, and the rest after the calculation is complete. The characters are examined in descending order of character reliability, so a suitable character is almost always available within the first 2 seconds. With a data matrix containing 582 characters and 584 taxa, about 320 characters are processed in this initial period, on a 133MHz Pentium computer.

'Best' algorithms should be able to handle numeric characters, as these often have high separating power. For example, the data set 'Festuca of North America' (Aiken et al. 1996) has 29 numeric characters and 67 multistate characters. When Intkey ranks these characters by their separating power, the top 17 characters are numeric. A similar tendency is shown in 'The Families of Flowering Plants' (Watson and Dallwitz 1992), which has 39 numeric characters and 459 multistate characters (excluding 'characters' used to define the classification). When the characters are ranked by separating power, 4 of the top 5, and 14 of the top 30, are numeric.

The high separating power of numeric characters is surprising to most taxonomists, as numeric characters are generally not very useful in conventional keys. There are two reasons for this. (1) Conventional keys must use multistate characters for numeric data, and this causes a loss of separating power. (2) Numeric characters often show a large amount of overlap between taxa; in conventional keys, this results in multiple occurrences of taxa, and an increase in the *printed* length of the key. Neither of these factors applies to interactive keys.

Separating a taxon

For confirming a tentative identification, it is useful to be able to rank characters according to how well they separate a given taxon from all the rest. Characters highly ranked on this criterion tend to lead to a shorter confirmation than those highly ranked on the ordinary 'best' criterion. 'Character reliabilities' (see below) should also be taken into account.

Best routes

Paths, similar to conventional keys, embedded in the interactive key. A path may be followed from the start of an identification; after it is left, ordinary interactive identification is resumed. This provides some guidance in the choice of characters. The method is inherently much less flexible than 'Best characters' (see above), and this limits its usefulness.

Differentiating attributes

These are attributes (i.e. character-state values) that are exhibited by only a small number of the remaining taxa — ideally 1.

This is not a desirable feature for identification, but it is included here because it is implemented in some programs.

We will confine our discussion to the case of differentiating attributes such that only a single one of the remaining taxa exhibits the attribute. The characters corresponding to these attributes tend to be the worst to use in identification, apart from those that do not differentiate the taxa at all or that differentiate only parts of taxa.

Differentiating attributes seem attractive because an appropriate one leads to an identification in a single step. The problem is the number of attributes that must be evaluated and rejected by the user before an appropriate one, that is, one exhibited by the current specimen, is found. For each attribute evaluated, there is a possibility that the user may *incorrectly* think that it describes the specimen; hence, the probability of error is increased more or less in proportion to the number of attributes evaluated. Furthermore, the chance of error is increased because the character states have to be evaluated in isolation, without reference to other states of the character.

In groups of 3 or more taxa, some may have no differentiating attributes at all, even though they are separable from the other taxa. For such taxa, the user has a pointless search through a potentially large list of attributes, with the possibility of error at each step. The best case for a group of n taxa is when each of the first n attributes in the list corresponds to a different taxon. In this case, the average number of steps required to complete the identification is $(1+2+\ldots n)/n=(n+1)/2$. This is $1/n$ *greater than* the *worst* case when 'Best characters' are used; that is, when the best available characters separate only one taxon from the rest, giving an average of $(1+2+\ldots n-1+n-1)/n=(n-1)(n+2)/(2n)$ steps to complete the identification. (The *best* case for 2-state characters, where the best available characters divide the remaining taxa into equal groups, requires about $\log_2 n$ steps.)

Here is an example using the sample data supplied with the DELTA programs (Dallwitz, Paine and Zurcher 1993a). Using the attribute 'leaf blades pseudopetiolate' eliminates all but 4 taxa: *Andropogon*, *Bambusa*, *Oryza*, and *Panicum*. In the list of differentiating attributes for these taxa, the position of the first differentiating attribute for each taxon is respectively 3rd, 2nd, 15th, and 1st. Thus the average number of attributes which must be examined to complete the identification correctly is (3+2+15+1)/4, that is, 5.25. Using the 'Best characters', the number of characters used is 2.

Removing redundant characters

It should be possible to remove from the list of available characters those that cannot separate the remaining taxa in an identification. This can and should be done as part of the 'Best characters' calculation (see above), but is included separately here because it is implemented separately in some programs.

Removing or flagging redundant character states

This is a way of preventing or inhibiting the selection of character states that are 'redundant', i.e., those that are not exhibited by any of the remaining taxa in an identification. It can be done either by removing the states entirely from the display, or by greying them. The program may or may not allow selection of the greyed states. Numeric characters can be treated similarly: by showing the range of values for the remaining taxa, and possibly by not allowing the user to enter values outside this range.

This is not a desirable feature for identification, but it is included here because it is implemented in some programs.

The intention of the feature is to assist the user by making the choice of states (or values) easier, because the choice is made from a restricted set. This may be slightly beneficial, provided that the first part of the identification is correct. However, if the user has made an error, if there is an error in the key's data, or if the specimen's taxon is not in the key, the user will be encouraged to enter values consistent with these errors, thus hindering their detection.

If the redundant states are not removed, selecting one of them will lead to the elimination of all taxa from the identification. This is a clear indication that an error has been made, and the program can guide the user in the resolution of the problem. For example, in these circumstances Intkey displays the following: the message 'No matching taxa remain'; a 'Help' button giving a full explanation of the possible causes of the problem, and information on how to proceed; and a button which, with a single click, increments the 'Error tolerance' (usually the best way to proceed).

If the redundant states are *greyed*, but may still be selected, the advantages described above are still available. However, the following problems arise. (1) The user may be puzzled by the greying (and users are reluctant to refer to the 'help' of interactive keys). (2) If the user does know the meaning of the greying, he may not know its significance to the identification process. (3) If the user does know the significance of the greying, his response may nevertheless tend to be biased in favour of the non-grey states, i.e., towards agreement with previously entered information. If some of the previous information is wrong (which is quite likely – see Effectiveness of Identification Methods – References), the new information is more likely to be wrong too, which makes it more difficult to recover, whether by manually correcting the errors or by an error-tolerance mechanism.

If the redundant states have been *removed*, and one of them is, in fact, exhibited by the specimen, the user requires considerable knowledge, acumen and alertness to recognize the situation, and to make use of the valuable information that it provides. Firstly, the pressure to accept one of the displayed states must be resisted. Secondly, the feature must be turned off (if possible) to check whether states have actually been removed, or whether the character simply does not cover all the possibilities. Thirdly, a way around the problem must be found, which might require searching the general 'Help' of the program.

In addition to, or instead of, greying the redundant states, the number of remaining taxa that exhibit (or may exhibit) each state may be displayed. The redundant states, if shown, will be associated with 0 taxa. This introduces another source of bias: the user will tend to select the states associated with smaller, but non-zero, numbers of taxa, because such states will tend to lead to a quicker identification.

Displaying the numbers of remaining taxa associated with states may be useful for information retrieval, i.e. for finding the numbers of remaining taxa recorded against each of the states. Note that the numbers must be calculated differently for information retrieval than for identification: in the former case, 'inapplicable' and 'unknown' values must be shown separately, and in the latter case, 'unknown' values must be counted against *every* state. Intkey has a separate option, 'Summary', for retrieval of this information.

Although removing redundant states is harmful for identification, removing redundant characters (those that cannot separate the remaining taxa) is beneficial. Choosing characters is fundamentally different from selecting states. The choice of a character can be unwise, but it cannot be right or wrong (provided that the program correctly implements character dependencies). However, a selected state can be right or wrong, i.e. exhibited or not exhibited by the specimen. Hiding redundant characters helps the user to

avoid unwise choices. Hiding or flagging redundant states may encourage the user to select a wrong state, because a hidden or flagged state may be the right one.

Redundant characters may, of course, contain information that is helpful in recovering from an error, or in increasing the user's confidence in an identification. To make the redundant characters available for these purposes, they may be made non-redundant by increasing the error tolerance (if this mechanism is available).

However, using redundant characters does not contribute towards reaching an identification. It is important to reach an identification quickly, even if the identification is wrong (or no taxa remain), because it is (usually) then that the user becomes aware that an error has occurred, and can take steps to overcome the problem.

For discussion, see DELTA-L: 'Hiding 'inapplicable' character states' 1995/2/2, and 'Comparison of interactive keys' 2006/3/22 to 2006/3/30. The latter thread presents a contrary view by Dmitry Dmitriev, the author of the interactive key-program 3I.

Character reliabilities

The 'reliability' of a character is a subjective measure, usually supplied by the author, of the character's accuracy and/or ease of use. The 'Best characters' algorithm (see above) should take into account both the separating power of a character, and its reliability. This is especially important with large character lists, so that the user does not have to skip over large numbers of inconvenient or unreliable characters when choosing a character. In interactive identification (in contrast to conventional keys), the reliability should be given only a small weight, so that the ranking of characters is mainly influenced by their separating power. This is because the user is in a position to judge how easily he can interpret a character in the particular specimen at hand, but cannot tell how well a character separates the remaining taxa.

Attribute reliabilities

The only advantage of a manually constructed conventional key over currently available interactive keys (or automatically generated conventional keys) is that the author, when deciding which character to use at a given point in the key, can take into account the reliabilities of the characters *for the taxa still in contention at that point*. Attribute reliabilities allow the same process in interactive keys (or in automatic generation of conventional keys, e.g. by the program Key (Dallwitz 1974)).

An 'attribute' is the value or values of a character for a particular taxon. An 'attribute reliability' is a subjective measure, supplied by the author, of the character's accuracy and/or ease of use for the taxon. It is best recorded as an increase or decrease in the overall or average reliability of the character (see 'Character reliabilities' above). Attribute reliabilities allow better performance of the 'Best characters' algorithm (see above).

For example, suppose that in a key to the species of several genera, the interpretation of a certain character is particularly difficult or error prone in genus *D*, and particularly easy and clear cut in genus *E*. If the attribute reliabilities are set to low values for species of *D* and to high values for species of *E*, the character will be placed lower in the 'Best' list while any *D* species are still in contention, and higher when the remaining species are mostly or only *E*.

It would also be useful to have an option for treating a taxon as 'variable' if its attribute-reliability was below a specified value, or was too far below the average value for the remaining taxa. Then the character could be safely used even when 'unreliable' for a few taxa.

Attribute reliabilities are not yet implemented in Intkey (or in Key). See Dallwitz, Paine and Zurcher (1993b, 1993c) for proposals for storing this information in DELTA format.

Searching the character list

It should be possible to search for text strings in the character list. Searching the states in addition to the 'features' should be optional.

Recording and matching character values

Retaining unknowns

Taxa for which a character is not recorded must be retained when that character is used (with any value) in an identification. This is essential for correct identification, but is not implemented in some programs.

In any program, ‘unrecorded’ or ‘unknown’ can be recorded as an ordinary character state. For example

#9. petals <colour>/
1. white/
2. red/
3. unknown/

This cannot take the place of a proper representation of ‘unknown’, and should never be done. If a taxon is recorded as ‘unknown’, i.e. ‘9,3’, it will be always be eliminated if the character is used in a key, i.e. when the user enters ‘9,1’ or ‘9,2’. The user will not enter ‘9,3’, because, unlike the author, he *does* know the colour of the petals – otherwise he would not use this character.

Character dependencies

The program should be aware that some values of certain ‘controlling’ characters make other ‘dependent’ characters inapplicable. When a controlling character is used in an identification, any characters that thereby become inapplicable should not appear in any lists of characters which are subsequently presented to the user.

Automatic setting of controlling characters

When a dependent character is used before its controlling character(s), the appropriate values of the latter should usually be set automatically. However, the author must have the option of overriding the automatic setting in cases when it would lead to incorrect or inconvenient results. In these cases, the user should be required to use the controlling character before the dependent character.

Automatic setting of the controlling character can lead to incorrect results when the controlling and dependent characters can convey conflicting information. For example, characters ‘leaf shape’ and ‘leaf length’ could be designated as being inapplicable when leaves are absent. If a user enters a leaf shape, or a non-zero value for leaf length, the program can correctly set ‘leaves present’. However, if a user enters 0 for the leaf length, then the intention is presumably to indicate that leaves are absent, and it would be wrong to automatically set ‘leaves present’. Therefore, the author should specify that when a user asks to use ‘leaf length’, the program should force the character ‘leaves (presence)’ to be used first.

It is also usually desirable for the user to set the value(s) of a controlling character when more than one value would be set automatically. For example, with the controlling character

#12. leaves <presence>/
1. well developed/
2. much reduced/
3. absent/

other leaf characters are inapplicable only if leaves are absent. Therefore, if one of these other characters is used (say, ‘leaves opposite’), a program might automatically set states 1 *and* 2: ‘leaves well developed or much reduced’. A particular specimen probably exhibits only one of these states, so discriminating power is lost unless the user changes character 12 to the observed value. It would be better to give the user the opportunity to set the observed value in the first place.

Gaps for integer numerics

The values for integer numeric characters should be able to contain gaps, for example, ‘5 or 10’ should be distinguishable from ‘5 to 10’. For example, the number of stamens can be used to distinguish Hyacinthaceae (3 or 6) from Lowiaceae (5).

Text characters

It should be possible to store free-text information about taxa, and to use this information for identification and information retrieval.

Special values for keys

When recording a taxon description, the author of a data set may record character-state values that are not, in fact, exhibited by the taxon, but may seem to be so, particularly to an inexperienced user. This nullifies the possible effect of the error on identifications, but has the disadvantages that the separating power of the character is reduced, and that the information is misleading in descriptions and for classification. It is therefore desirable that the author should be able to flag such values, and that the user should have the option of ignoring them.

Intkey does not currently offer this feature, as it is not supported in the DELTA format from which the Intkey data are derived. The feature is among the enhancements that will be supported in the new DELTA system (Dallwitz, Paine and Zurcher 1993b, 1993c).

Probabilistic identification

The data ‘matrix’ should have provision for storing the probabilities that a given taxon will exhibit certain character states, and the program should be able to make use of these probabilities in the identification process — see, for example, Willcox and Lapage (1975), Bryant (1998).

The probabilities that the user will make errors in assigning character states to the specimen are often substantial (Stucky 1984; Fermanian et al. 1989; Morse et al. 1996; Tardivel and Morse 1998), and these probabilities need to be taken into account in probabilistic identification, even for deterministic attributes (that is, where the probabilities of the state values are recorded as 1 or 0 in the data matrix). Willcox and Lapage (1975) describe the difficulties of dealing with errors in the context of identification of bacteria, and the problem is likely to be quantitatively worse in general biological identification, because characters are less standardised, and the users of the keys are likely to vary more in their knowledge and experience.

There are other potential impediments to the widespread use of probabilistic identification. Probabilities are not widely available in the literature, and taxonomists might be reluctant to take the trouble to estimate and record them. Similar problems are already evident in the relatively infrequent use of numerical values compared with terms such as ‘small’, ‘large’, ‘sparse’, and ‘dense’. Also, there will be greater difficulty for the user in understanding the processes and the results.

Probabilities cannot be recorded in the current DELTA format, except as comments, but they will be recordable in the enhanced DELTA system (Dallwitz, Paine and Zurcher 1993b, 1993c). This will open the way for their use in Intkey.

Inapplicable versus unknown

Inapplicable values, including those not resulting from character dependencies, should be distinguished from unknown values.

Expanded ranges for numeric characters

It should be possible, for the purpose of identification, to expand the range of values recorded for a numeric character that has been poorly sampled in a taxon.

This is possible to a limited extent in Intkey: single recorded numeric values can be transformed into ranges when the original data are converted from DELTA format into the format used by Intkey. This process should preferably be under the control of the user of the interactive key, so that the size of the range can be varied, and so that the actual recorded data are available for information retrieval.

Unknown state values

It should be possible to record individual state values of multistate characters (as opposed to the character as a whole) as unknown. Most interactive identification programs allow a character to be scored ‘unknown’ for a taxon. For multistate characters, the information from a single specimen is usually assumed to be typical of the taxon as a whole. Thus, the observed values are recorded against the taxon, and it is assumed that the other values are not present in the taxon. Although this is usually satisfactory, it should also be possible to record as ‘unknown’ those states that have not actually been observed. This is particularly important for characters such as geographical distribution, and the host range of parasites.

This feature is not currently available in Intkey, but will be available in the enhanced DELTA system now under development (Dallwitz, Paine and Zurcher 1993b, 1993c).

Exact characters

‘Exact’ characters are ones that are to be regarded as not subject to error. When an ‘exact’ character is used in identification or queries, taxa inconsistent with the specified value are eliminated, regardless of the setting of the ‘Error tolerance’ (see above).

For example, in a key to molluscs (Ponder, Clark and Dallwitz 2000), the first character is

#1. <Major group:>/
1. gastropod/
2. bivalve/
3. cephalopod/
4. chiton/

Most of the other characters are applicable to only one of these major groups. Thus, character 1 is usually the only character separating, say, a gastropod species from a bivalve species, and it is normally the first one used in an identification. Suppose that an attempt is made to identify a gastropod species, but the identification is found to be incorrect. If the error tolerance is increased, most or all of the bivalve, cephalopod, and chiton species will come back into contention, but there will be no characters available to separate them from the gastropods. This problem is overcome by making character 1 ‘exact’; then increasing the error tolerance brings only gastropod species back into contention.

Fixing character values

It is convenient to be able to specify character values that are not to be cleared when a new identification is started. For example, if there is a character whose states specify geographical regions, and several specimens coming from the same region are to be identified, then the appropriate state value need be entered only once.

Subsets

Named subsets of characters

There should be a mechanism for naming subsets of the characters, so that they can be easily found and used. The mechanism should be available to the user, not just to the author.

Global subsets of characters

It should be possible to specify subsets of characters to which all subsequent operations will be restricted. For example, the characters could be restricted to those available in an incomplete specimen.

In Intkey, the restriction may be carried out by either inclusion or exclusion of specified characters.

Local subsets of characters

It should be possible to specify a subset of characters to be used for a single operation.

Named subsets of taxa

There should be a mechanism for naming subsets of the taxa, so that they can be easily found and used. The mechanism should be available to the user, not just to the author.

Global subsets of taxa

It should be possible to specify subsets of taxa to which all subsequent operations will be restricted. For example, the taxa might be restricted to those known to occur in a particular geographical area.

In Intkey, the restriction may be carried out by either inclusion or exclusion of specified taxa.

Local subsets of taxa

It should be possible to specify a subset of taxa to be used for a single operation.

Character interpretation

Character notes

Extensive text to aid interpretation of characters should be conveniently available within the system. (The notes might also include information about the reasons for defining characters in particular ways.)

Glossaries

It should be possible to link words, in any context in the database, to their definitions.

This facility is not available in Intkey (the information can be provide in character notes, but their use for this purpose tends to be clumsy and repetitive).

Character illustrations

It should be possible to display illustrations of characters.

State selection from illustrations

It should be possible to select values from the character-illustration screens during identification.

Images and sounds

Taxon illustrations

It should be possible to display illustrations of taxa.

Taxon illustrations by subject

It should be possible to select illustrations of taxa by subject (for example, flowers, habitat, distribution map).

Flexible display of illustrations

Illustrations of any size should be able to be scaled, scrolled, repositioned, displayed simultaneously, and tiled or cascaded. It is convenient if all illustrations can be closed with a single operation.

Text with illustrations.

Text should be able to be displayed with illustrations (instead of being built into them). This facilitates maintenance, ensures that text remains legible when an illustration is scaled, and allows the use of multiple languages with a single illustration.

Sounds

It should be possible to associate sounds with characters and taxa.

Videos

It should be possible to associate videos with characters and taxa.

Running without illustrations

It is desirable for a package normally containing illustrations to be able to run well without them. For example, there may not be enough disk space for the illustrations, or network access to the illustrations may be too slow.

Linked keys

Integral hierarchical keys

It should be possible to have keys for different taxonomic levels (for example, keys to species and genera) within a single data matrix. It should be possible to change the level during an identification without losing information.

In Intkey, this is achieved simply by restricting the view to the taxa of the desired level, via the general mechanisms for handling subsets of taxa (see ‘Subsets’ above). If an identification is in progress, the information previously entered about the specimen is re-applied to the new set of taxa.

Separate hierarchical keys

It should be possible to link taxa to other keys, to allow identification to proceed to a lower taxonomic level. For example, a genus could be linked to a key to the species of the genus.

Intkey allows taxa to be linked to any files that are associated with an application to process them. If the linked file is another Intkey data set (that is, another key), Intkey itself processes the linked file. The link could also be to a conventional key as plain text, RTF, or HTML.

Information retrieval

Searching the taxon names

It should be possible to search for text in the taxon names and, optionally, the synonyms and common names.

Descriptions from the data

It should be possible to display descriptions generated from the data used in identification. Some programs can display only text descriptions that have been entered independently of the data matrix.

Differences between taxa

The program should be able to find the differences between members of a set of taxa. There should be no restrictions on the size of the set of taxa.

Similarities between taxa

The program should be able to find the similarities between members of a set of taxa. There should be no restrictions on the size of the set of taxa.

Diagnostic descriptions

The program should be able to find diagnostic descriptions, which distinguish a given taxon from all the other taxa. These provide a quick way of confirming the identity of a specimen. The characters should be chosen from those that have not been used in the current identification, in order to provide an independent confirmation.

Intkey has a parameter, ‘DiagLevel’, which specifies the minimum number of characters for which the diagnostic description should differ from all the other taxa. Another parameter, ‘DiagType’, distinguishes between specimen-diagnostic and taxon-diagnostic descriptions. The latter are allowed to contain characters that may sometimes be inapplicable to specimens belonging to the taxon.

Taxon retrieval by attributes

It should be possible to find all taxa that have certain attributes or combinations of attributes. It must be possible to eliminate taxa for which the attribute is unknown or inapplicable.

Control of value matching

The user should have flexible control over which character values or combinations of values are to be regarded as equal to other values.

In Intkey, matching in identification and in the Differences and Similarities commands is controlled by the Set Match command, which has the options ‘Inapplicables’, ‘Unknowns’, ‘Subset’, ‘Overlap’, and ‘Exact’. ‘Set Match Inapplicables’ and ‘Set Match Unknowns’ specify respectively that ‘inapplicable’ and ‘unknown’ values match any value. ‘Set Match Subsets’ specifies that two sets of values match if one set (usually the values of the specimen) is a subset of the other (for example, ‘1 or 2’ matches ‘1 or 2 or 4’ but not ‘2’ or ‘3’; ‘2–5’ matches ‘1–6’ but not ‘4–10’). ‘Set Match Overlap’ specifies that two sets of values match if they overlap, that is, if they have any values in common (for example, ‘1 or 2’ matches ‘2 or 3’; ‘2–5’ matches ‘4–10’). ‘Set Match Exact’ specifies that two sets of values match only if they are identical. The default setting is ‘Overlap Unknowns Inapplicables’, which is usually the most appropriate for identification. For information retrieval, the most appropriate setting is usually ‘Overlap’.

Character-value distributions

It should be possible to display the distribution of character values within a set of taxa.

Most similar taxa

It should be possible to rank taxa by their similarity to a given taxon.

Intkey does not have this facility, but it is available in other programs in the DELTA System.

Text files attached to taxa

It should be possible to link a taxon to text files containing information about the taxon.

Intkey allows linking to any text file for which a viewer is available, for example, MS-Word documents or HTML files (including files on the Internet). Intkey itself handles the viewing of RTF files.

Data sharing

Importing DELTA format

The DELTA (DEscription Language for TAxonomy) data format (Dallwitz 1980; Dallwitz and Paine 1993) has been endorsed by the International Taxonomic Databases Working Group as a standard for interchange of descriptive data. It should be possible to create an interactive key from DELTA data.

Exporting DELTA format

The ability to export DELTA data from the interactive key, or from the system that generates the key, makes the data available to various programs that support DELTA format.

Importing SDD format

The SDD (Structured Descriptive Data) data format (Hagedorn 2005) has been endorsed by the International Taxonomic Databases Working Group as a standard for interchange of descriptive data. It should be possible to create an interactive key from SDD data.

Exporting SDD format

The ability to export SDD data from the interactive key, or from the system that generates the key, makes the data available to various programs that support SDD format.

Data output

It is useful to be able to output program results in forms suitable for input to other programs.

For example, Intkey can output lists of taxon numbers for taxa having specified attributes, DELTA descriptions of individual taxa, combined DELTA descriptions of groups of taxa, diagnostic characters, differences and similarities between taxa.

Links with description writing

It should be possible to generate publication-quality descriptions from the same data that are used to construct the identification system. High-quality descriptions require provision for text characters, and for qualifying comments associated with the recorded values of multistate and numeric characters.

Links with key generation

It should be possible to generate conventional keys from the same data that are used to construct the identification system.

Links with classification

It should be possible to carry out cladistic and phenetic analyses from the same data that are used to construct the identification system. This generally requires a mechanism for excluding characters that are not suitable for use in classification.

Useability

Online help

The program should have complete, context-sensitive, built-in help.

Command files or macros

There should be a mechanism for storing and repeating a series of operations.

User-definable toolbar

The author or user of a dataset should be able to define toolbar buttons for easy access to commonly used operations or groups of operations.

External program text

The program text (commands, help, messages, etc.) should be external to the program, allowing easy creation and use of different language versions.

Log files

It should be possible to create a file showing the history (input and output) of a session.

Unlimited data size

There should be no arbitrary limits on numbers of taxa, characters, or character states.

Unlimited field lengths

There should be no arbitrary limits on lengths of taxon names, text of characters and character notes, or text fields in taxon descriptions.

No special memory requirements

The program should be able to run with the minimum amount of memory normally needed to run the operating system.

Fast execution

Execution times for operations that are frequently used in identification should be reasonably small with large data sets.

Most programs are sufficiently fast when handling operations involving only a single character or taxon, for example, when using a character in an identification. (However, programs that require an Internet

transaction for such operations are limited by the response time of the network – see Dallwitz, Paine, and Zurcher (2002).)

The most time-consuming operations involve scanning large parts of the data matrix. Examples are the calculations of the 'Best characters' in an identification (see above), the differences between large numbers of taxa, and 'Diagnostic descriptions' (see above).

Internet capability

The program should be able to access data, images, and other information over the Internet.

Intkey downloads its main data files in compressed form at the start of a session, and they may be saved on the local computer for future use (in addition to the caching provided by the operating system). Character and taxon images, and other files attached to taxa, are downloaded as required.

Installation unnecessary

It is convenient to be able to run the program directly from a CD-ROM, without installation.

Intkey packages on CD do not require installation. Having the package on the hard disk does improve the execution speed, and this can be done simply by copying the complete directory tree to the hard disk (and undone simply by deleting the tree). It is also easy to put some of the package on the hard disk, and access bulky and less frequently used parts, such as the taxon images, from the CD.

Simple user interface

The user interface should be simple and efficient, and should provide an easy transition from use of basic features to use of the full functionality.

Intkey is suitable for users ranging from primary-school children to professional taxonomists, who will find it useful for many aspects of taxonomic work, not just identification (e.g. Aiken *et al.*, 1997). There are two modes of operation: Normal and Advanced. In Normal mode, the program is operated via the toolbars. Some options are not normally available (although any option can be made available by adding it to the custom toolbar), and most of the menu system and many dialog-box buttons are hidden. In Advanced mode, all options are available, and the program can be operated via the toolbars, the menu system, or a command line.

Construction of interactive keys

Developing a character list: reproducibility

If you are starting a dataset from scratch, begin by entering only a few characters and a few taxa (say about 5 of each), and recording the data for these taxa. Then test *all* the applications you intend to use — for example, produce natural-language descriptions, a conventional key, an interactive key, and a cladistic tree. Then add a few more characters and taxa, and repeat the testing. This iterative procedure helps you develop strategies for character definition (e.g. wording, number of states, multistate versus numeric, arrangement in sentences), which you can apply and refine as development of the character list proceeds.

Ideally, after the above process, the character list should be tested by having several people independently record descriptions of about 10 disparate taxa. The different versions of the descriptions will probably be different, i.e. the results will not be reproducible. You need to detect such problems before you have recorded much data.

The Differences option in Intkey can easily pinpoint the discrepancies.The reasons for them can then be discussed, and the character list refined on this basis. This process should then be iterated, adding a few more taxa each time, until the character list seems satisfactory.

Character notes, and preferably character illustrations, should be added during the development of the character list. There is a tendency to think that these are only for the benefit of the end user, and to postpone adding them. However, they are an essential part of obtaining reproducible descriptions. Without precise character notes, the character concepts of even the principal author may drift during the recording of the data, as more specimens are seen. Then, data recorded near the end of the project may not be comparable to data recorded near the beginning. Pay particular attention to the boundaries between character states, i.e. describe and illustrate intermediate conditions, and say to which state they are deemed to belong (unless you intend recording such intermediates as belonging to both states).

You should also record your *reasons* for defining the characters as you have, while these reasons are still fresh in your mind. Otherwise, other taxonomists (or even you yourself, later) may fail to fully appreciate the definitions, and change them, to the detriment of your data and possibly to the detriment of future data recording.

Redundancy in the data

To fully realise the advantages of interactive keys, the data must have a considerable amount of redundancy. Otherwise, the user will have limited choice of characters, and the error-tolerance capability will not work well, if at all. In particular, if the information in the data matrix is obtained solely from a conventional key, then the characters that would be used to identify a given specimen with the conventional key will also have to be used to identify the specimen with the interactive key.

Intkey provides an easy method of checking the data matrix for redundancy: the command ‘Diagnose All’ is run with a high setting of the DiagLevel parameter, e.g. 5. The program generates diagnostic descriptions for all the taxa, and displays lists of those that are not separable at the requested level. These lists show the actual number of differences between each taxon and the taxon being diagnosed. It is easy to see, from this information, where the data need to be strengthened by filling in missing information or by adding extra characters. If extra characters are added, they need only be recorded for the poorly separated taxa.

Keying directly to species

Large conventional keys are difficult to construct, so hierarchies of smaller keys are mostly used instead. For example, an identification might be carried out by first using a key to families, then a key to the genera of a family, then a key to the species of a genus. As a good interactive key program can easily cope with many thousands of taxa, it is usually possible to eliminate some of these steps, for example, to construct a single key to all the species of a family. There are three advantages in doing so.

1. The efficiency of the key is not hampered by uneven distribution of species among the higher taxa. For example, consider 8 species belonging to 4 genera that contain 4, 2, 1, and 1 species. Using 2-state characters, the shortest key directly to species contains 3 steps for each species. The shortest key to genera contains 2 steps for each genus; keying these out to species requires 2, 1, 0, and 0 extra steps, respectively, giving an average of 3.25 steps per species.

2. Information loss in going from one key to another is avoided. For example, supposing an identification to genus level uses a character that varies between species of the genus. The possible identity of the specimen has then been restricted to a subset of the species of the genus. However, at the start of the key to the species, all species of the genus are once again in contention (unless the relevant characters are available in both keys, and the corresponding attributes of the specimen are carried automatically from the first key to the second).

3. All characters used in a single interactive key are subject to the error-tolerance mechanism, whereas an error made in a higher-level key cannot usually be corrected once a lower-level key is in use.

One advantage of a hierarchical set of keys is that it can be entered at any level. This facility can easily be incorporated in a single interactive key by including ‘characters’ giving the higher taxa to which the species belong. This approach has the advantage that, because of error tolerance, a user’s error in assigning a specimen to a certain high taxon does not inevitably lead to a wrong identification.

Similarly, including ‘characters’ for geographical distribution gives most of the benefit of special regional keys, plus the advantage conferred by error tolerance if the specimen is the first record of its taxon in a region.

Strategies for using interactive keys

‘What is the first business of him who philosophizes? To throw away self-conceit. For it is impossible for a man to begin to learn that which he thinks that he knows.’

> Epictetus — The Discourses, Book II, Ch. 17 – How we must adapt preconceptions to particular cases (How to apply general principles to particular cases)

The flexibility of interactive keys, which is one of their major advantages, can also lead to poor results if appropriate strategies for their use are not adopted.

The goal is to maximize the probability of obtaining a correct identification. This probability is equal to the product, over all the characters used, of the probabilities of correctly applying each character to the specimen. So the user should aim to choose characters for which there is a low probability of making an error, *and* which will tend to minimize the number of characters required to complete the identification.

The character reliabilities set by the author will give some guidance in choosing characters with a low error probability, and they influence the order in which the characters are ranked in the ‘Best’ list in Intkey. However, the reliability is set for an average user and an average taxon. Even an inexperienced user can improve on this by evaluating how easily he is able to apply the character to the specimen at hand.

Minimization of the number of characters used is achieved by using characters that eliminate as many taxa as possible. For the user who is not familiar with the taxa, this means choosing characters near the top of the list of ‘Best’ characters. In Intkey’s Advanced mode, more precise guidance is provided by displaying the ‘separating power’ of the characters. The user who is familiar with the taxa may be able to improve on this by recognising characters for which the specimen exhibits a state that is unusual in the group of taxa still under consideration, but this is difficult during the later stages of an identification.

In conventional keys, if the user is uncertain which character state applies to the specimen there is usually no option but to guess. This is a poor strategy in interactive keys. Rather, the character should be avoided. Alternatively, if the character has more than two states it may be possible to reduce the probability of error by selecting more than one state.

A strategy often used by people new to interactive keys, and worth mentioning so that it can be avoided, is to use each character, in the order in which they are stored in the key, until a result is obtained. This violates all of the above principles.

The result of any identification should be regarded as a hypothesis requiring confirmation. Intkey provides several ways of doing this. The specimen can be compared with illustrations of its putative taxon, with a full description, or with a diagnostic description employing characters not already used in the identification. Alternatively, the error tolerance can be increased, bringing previously eliminated taxa back into contention, and the identification process continued. The ‘Separate’ option, available in Intkey’s Advanced mode, should be used in preference to ‘Best’ for this purpose, because it ranks characters according to their ability to separate a particular, rather than an average, taxon from the rest.

When testing an interactive-key program, it is tempting to *pretend* to do an identification — that is, to go through the motions of an identification when you already know the answer. This is not generally a good test of the value of a program for real identification. Ideally, the test should be carried out in a group with which you are unfamiliar (the more so, the better, for example, in a Kingdom different from your speciality). If this is not possible, pay special attention to the easy availability of features which help an inexperienced user: ‘best’ characters, character illustrations and notes, error tolerance, and expressing uncertainty by selecting more than one state.

History of interactive keys

The table below shows when various features of interactive keys were introduced. The features shown in bold are those most important for efficient and accurate identification (Dallwitz 2000a).

Publication/program	Features important for identification
Goodall (1968).	**Unrestricted character use, numeric characters, expressing uncertainty, retaining unknowns**.
Morse (1971).	**Error tolerance, best characters**.
Pankhurst & Aitchison (1975). Online V5, 1987.	Character deletion and changing, character dependencies, differences between taxa, separating a taxon.
Watson, Gibbs Russell, and Dallwitz, (1989). Intkey V2.	Locating errors, character reliabilities, subsets of characters and taxa, **gaps for integer numerics**, diagnostic descriptions.
Intkey V4, 1995.	Character and taxon images, taxon images by subject, character notes, state selection from images, flexible display of images, searching the character list.

References

Aiken, S. G., Dallwitz, M. J., McJannet, C. L., and Consaul, L. L. 1996 onwards. *Festuca* of North America: descriptions, illustrations, identification and information retrieval. http://nature.ca/aaflora/data/

Aiken, S.G., Dallwitz, M.J., McJannet, C.L. and Consaul, L.L. 1997. Biodiversity among *Festuca* (Poaceae) in North America: diagnostic evidence from DELTA and clustering programs, and an INTKEY package for interactive, illustrated identification and information retrieval. *Canadian Journal of Botany* 75: 1527–1555.

Bryant, T. N. 1998. Probabilistic identification systems for bacteria. In 'Information technology, plant pathology and biodiversity', pp. 315–332. (Eds P. Bridge, P. Jeffries, D. R. Morse, and P. R. Scott.) (CAB International: Wallingford.)

Dallwitz, M.J. 1974. A flexible computer program for generating identification keys. *Systematic Zoology* 23: 50–57.

Dallwitz, M.J. 1980. A general system for coding taxonomic descriptions. *Taxon* 29, 41–46.

Dallwitz, M.J. 1992. A comparison of matrix-based taxonomic identification systems with rule-based systems. In 'Proceedings of IFAC workshop on expert systems in agriculture', pp. 215–218. (Ed. Xiong, F.L.) (International Academic Publishers, Beijing.) Also available at http://delta-intkey.com

Dallwitz, M. J. 1993. DELTA and INTKEY. In 'Advances in computer methods for systematic biology: artificial intelligence, databases, computer vision', pp. 287–296. (Ed. R. Fortuner.) (The Johns Hopkins University Press: Baltimore, Maryland.)

Dallwitz, M.J. 1994. Desirable attributes for interactive identification programs. http://usobi.org/archives/taxacom.html

Dallwitz, M.J. 1996 onwards. Programs for interactive identification and information retrieval. http://delta-intkey.com

Dallwitz, M. J. 1997 onwards. Desirable attributes for interactive identification programs. http://delta-intkey.com

Dallwitz, M.J. 1999 onwards. A comparison of formats for descriptive data. http://delta-intkey.com

Dallwitz, M. J. 2000a onwards. A comparison of interactive identification programs. http://delta-intkey.com

Dallwitz, M.J. 2000b onwards. Data requirements for natural-language descriptions and identification. http://delta-intkey.com

Dallwitz, M.J. and Paine, T.A. 1993 onwards. Definition of the DELTA format. http://delta-intkey.com/

Dallwitz, M.J., Paine, T.A. and Zurcher, E.J. 1993a onwards. User's guide to the DELTA system: a general system for processing taxonomic descriptions. http://delta-intkey.com

Dallwitz, M.J., Paine, T.A. and Zurcher, E.J. 1993b onwards. New features for the DELTA system. http://delta-intkey.com/

Dallwitz, M. J., Paine, T. A., and Zurcher, E. J. 1993c. Preliminary suggestions for new features for the DELTA system. *DELTA Newsletter* 9: 2–13.

Dallwitz, M. J., Paine, T. A., and Zurcher, E. J. 1995 onwards. User's guide to Intkey: a program for interactive identification and information retrieval. 1st edition. http://delta-intkey.com

Dallwitz, M. J., Paine, T. A., and Zurcher, E. J. 1998. Interactive keys. In 'Information technology, plant pathology and biodiversity', pp. 201–212. (Eds P. Bridge, P. Jeffries, D. R. Morse, and P. R. Scott.) (CAB International: Wallingford.)

Dallwitz, M. J., Paine, T. A., and Zurcher, E. J. 2002 onwards. Interactive identification using the Internet. http://delta-intkey.com

Edwards, M. and Morse, D.R. 1995. The potential for computer-aided identification in biodiversity research. *Trends in Ecology and Evolution* 10: 153–158.

Fermanian, T.W., Barkworth, M. and Liu, H. 1989. Trained and untrained individual's ability to identify morphological characters of immature grasses. *Agronomy Journal* 81, 918–922.

Goodall, D. W. 1968. Identification by computer. *Bioscience* 18: 485–488.

Hagedorn, G.; Thiele, K.; Morris, R. and Heidorn, P. B. 2005 onwards. The Structured Descriptive Data (SDD) w3c-xml-schema, version 1.0. http://www.tdwg.org/standards/116/

Morse, D.R., Tardivel, G.M. and Spicer, J.I. 1996. A comparison of the effectiveness of a dichotomous key and a multiaccess key to woodlice. Technical Report 14–96, Computing Laboratory, University of Kent at Canterbury, UK.

Morse, L. E. 1971. Specimen identification and key construction with time-sharing computers. *Taxon* 20: 269–82.

Pankhurst, R.J. 1991. Practical taxonomic computing. 202pp. (Cambridge University Press, Cambridge.)

Pankhurst, R.J. and Aitchison, R.R. 1975. An on-line identification program. In 'Biological Identification with Computers', pp. 181–185. (Ed. R. J. Pankhurst.) (Academic Press, London.)

Ponder, W.F., Clark, S.A., and Dallwitz, M.J. 2000. Freshwater and estuarine molluscs: an interactive, illustrated key for New South Wales. CD-ROM. (CSIRO Publishing, Melbourne.)

Stucky, J.M. 1984. Comparison of 2 methods of identifying weed seedlings. *Weed Science* 32: 598–602.

Tardivel, G. M., and Morse, D. R. 1998. The role of the user in computer-based species identification. In 'Information technology, plant pathology and biodiversity', pp. 247–259. (Eds P. Bridge, P. Jeffries, D. R. Morse, and P. R. Scott.) (CAB International: Wallingford.)

Watson, L., and Dallwitz, M. J. 1992 onwards. The families of flowering plants: descriptions, illustrations, identification, and information retrieval. http://delta-intkey.com

Watson, L., Gibbs Russell, G.E. and Dallwitz, M.J. 1989. Grass genera of southern Africa: interactive identification and information retrieval from an automated data bank. *South African Journal of Botany* 55: 452–63.

Willcox, W.R. and Lapage, S.P. 1975. Methods used in a program for computer-aided identification of bacteria. In 'Biological Identification with Computers', pp. 103–119. (Ed. R. J. Pankhurst.) (Academic Press, London.)