7 AUTOMATING THE INEFFABLE:

Qualitative software and the meaning of qualitative research

Nigel G. Fielding

Computers have been used for half a century in research using textual data but the introduction of software to support qualitative analysis has proven controversial in a way that it never was in the field of content analysis. To understand this controversy, and its implications for qualitative research, we begin with a description of qualitative software. We note that for most users the story of qualitative software is the story of the analytic procedure called 'code-and-retrieve'. But we will also sketch in other promising applications of qualitative software, including approaches to data analysis which are implicitly numerical or are based on formal logic.

The chapter moves on to explore how people actually use qualitative software, demystifying what sometimes seems a technical 'black art'. The point will be made that patterns of use are as apt to reflect the research environment as the technology. But we will also observe that software could play a significant part in changing the workings of the qualitative research community. However, the epistemological implications of qualitative software do not mirror the extent of its technical, practical and research environment implications. Epistemological preoccupations are more enduring than any technology.

WHAT IS COMPUTER ANALYSIS OF QUALITATIVE DATA?

Computers have already established a considerable presence in qualitative research, despite its self-image as a craft. Computers are routinely used at each stage of qualitative projects, from data collection through analysis to presentation (Weitzman and Miles 1995). Yet the information technologies which now support our writing, data collection and literature searches have attracted little comment, in contrast to the debates prompted by the emergence of software to support qualitative data analysis. It may not be coincidental that, in a field which has been slow to document and codify its procedures, the last practice to attract such attention has been that of data

analysis. Indeed, even those indifferent to qualitative software may concede that its emergence has benefited the field by obliging us to be more explicit about how we manage, analyse and interpret qualitative data.

A dominant concern about qualitative software is that it may somehow 'take over' the analysis, imposing a standard approach and employing concealed assumptions, the so-called 'Frankenstein's monster' debate (Lee and Fielding 1991; Kelle 1995). It is sometimes forgotten that the monster was a threat only because he was abused by humankind. Qualitative software is, even now, quite limited in the kinds of support it offers for analysis, and there is no prospect that it will ever excuse the need for researchers to think. There is another kind of rebuttal to the take-over fear, though. The imposition of a standard approach is obviated by the sheer variety of packages, each displaying considerable distinctiveness. Each has merits and deficiencies which suit it better for particular kinds of work. Researchers need to determine which best fits the kind of work they do, both practically and analytically.

Software for computer-assisted qualitative data analysis (CAQDAS) seeks to facilitate data management chores which are tedious and subject to error when done manually, make the analytic process more 'transparent' and accountable, and support analytic approaches which would otherwise be cumbersome. From the late 1970s, when the first packages emerged for use on mainframe computers, the notable thing about qualitative software was that it was developed by social scientists, not computer scientists. Development was often in response to data management or analysis requirements on specific projects. The prototypes may have suffered bugs and forbidding interfaces but they were closely informed by what researchers regarded as essential to support qualitative data analysis. Although commercial pressures inevitably intrude as the software becomes more professional, developers still exchange ideas via conferences and publications, and most remain social scientists as well as developers. This makes for close awareness of developments with other packages and responsiveness to the needs of users.

The earliest computer application to textual data was content analysis, which pursues 'objective, systematic and quantitative description of the manifest content of communication' (Berelson 1952: 489). Programs had to store extensive documentary data, address text units as 'variables', produce 'concordances' (organized lists of words and phrases) and quickly access the context of particular text segments. The mode of analysis meant programs required statistical features or to allow data to be exported to statistical packages. These requirements produced software of limited use to qualitative researchers.

To marshal the variety of data that qualitative researchers use, software must be able to store data in different formats and enable researchers to annotate it so as to track the effects of field relations on it. Researchers need to move around the corpus looking at the relevance of specific analytic ideas to different segments, so they require a means to 'navigate' the database.

They must also be able to retrace their steps in these operations. 'Coding' is fundamental to qualitative data analysis. The corpus has to be divided into segments and these segments assigned codes (or categories) which relate to the analytic themes being developed (similarly, ethnomethodologists and conversation analysts need to assign to segments a term for routines which recur in behaviour/talk). Researchers aim for codes which capture some essential quality of the segment, and which apply to other segments too. Otherwise every segment would be a code and we would get nowhere; data analysis is a process of data reduction. During this process we sometimes need to revise codes, either globally or in application to particular segments. When data are coded we can retrieve them selectively using various analytic strategies. Researchers also need to be able to write 'analytic memos', for example, giving the reasons why they assigned a given code to a segment or part containing the germ of an analytic point they want to develop.

A TYPOLOGY OF QUALITATIVE SOFTWARE

Coding and retrieval of coded segments informs all three basic types of qualitative software: *text retrievers, code-and-retrieve* packages and *theory-building* software (Weitzman and Miles 1995). *Text retrievers*, such as Metamorph, WordCruncher, ZyINDEX and Sonar Professional, let users recover the data pertaining to each category where keywords appear in the data. For example, you may have a category called 'social class'. When you search for 'social class', wherever these words appear in the text the software will extract it. If the respondent did not use the keyword it must be added to the data so the segment can be retrieved.

Words, other character strings and combinations of these, in one or many files, can be retrieved, along with things that sound alike, mean the same thing, or have patterns like the sequences of letters and numbers in social security records. Retrieved text can be sorted into new files, and analytic memos can be linked to the data. Content-analysis capabilities are often provided, including facilities to handle quantitative data. Sophisticated variants like askSam and FolioVIEWS have more ways to organize text and make subsets. Some deal with highly structured text organized into 'records' (specific cases) and 'fields' (numerical or text information appearing for each case). Text retrievers are fast at retrieval. Their forte is information management where basic data is held on large numbers of people.

Code-and-retrieve packages, including HyperQual, Kwalitan, WinMAX and The Ethnograph support the division of text into segments, attaching codes to the segments, and retrieving segments by code (or combination of codes). A 'single sort' retrieves all the data pertaining to one code. 'Multiple sort' retrievals handle cases where one category is considered in relation to another, for example where data coded 'age' coincides with data coded 'social class'. Users may also do searches which recover only data where two particular characteristics apply but not a third; for instance, data from MALE

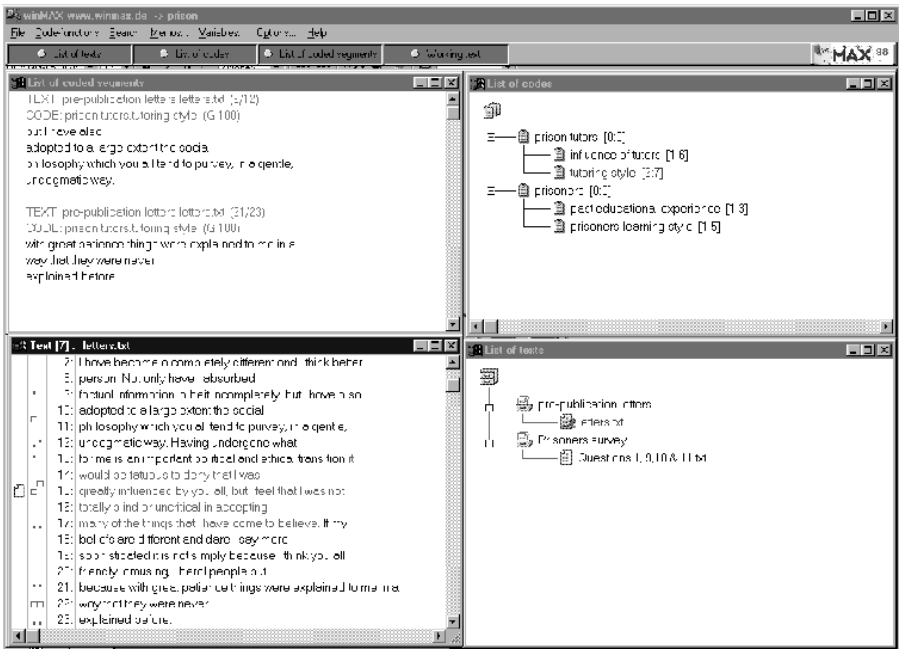

FIGURE 7.1 EXAMPLE OF CODE AND RETRIEVE SOFTWARE

respondents living in URBAN areas who are NOT married. Such strategies are called Boolean retrievals (following the 'and', 'or' and 'not' relations of Boolean mathematics). Various types of Boolean retrieval are supported.

Code-and-retrieve packages approach the writing of memos in different ways. Ideally the software reminds users that an analytic memo pertains to a segment every time it is inspected and displays the memo instantly if desired. Researchers sometimes need to retrace their interpretive work, particularly in team research, for example, to check coding has been consistent. Hypertext features allow users to move quickly around (navigate) the database.

WinMAX offers an example of contemporary code-and-retrieve software (Figure 7.1). The window at top right lists codewords hierarchically, with main codes followed by sub-codes; 'tutoring style' is highlighted as it is the code currently being considered. The bottom right window is a hierarchical list of the texts being analysed. At bottom left is the window containing the raw data currently being worked on, an extract from correspondence with a prisoner about the prison education programme. Note the icon in the extreme left margin; it indicates a memo has been written relating to a segment of the extract; to see it, the user clicks on the icon. In the next margin are markers indicating the extent of segments of the extract to which codes have been applied. The top left window lists the segments pertaining to the 'tutoring style' code, retrieved so the user can compare them.

Such software focuses analytic attention on relationships between codes (or categories) and data. *Theory-building software*, such as Atlas/ti, HyperRESEARCH, NUD*IST4, and N-vivo, emphasizes relationships between the categories, although code-and-retrieve support is also offered. These packages help users develop higher-order classifications and categories than those derived directly from data, formulate propositions which fit the data and test their applicability, or visualise connections between categories as an aid to conceptualization. Full Boolean searching and the ability to 'test' hypotheses may be offered. Some can show code names (or other objects, like memos) as nodes in a graphic display and users can link them to other nodes by specified relationships like 'leads to', 'is a kind of' and so on.

Such software can facilitate formal approaches to qualitative data analysis. A controversial example is the 'hypothesis test feature' in HyperRESEARCH. Users hypothesize a relationship between the occurrence of a particular statement in the data and the occurrence of another (*if* 'this' *then* 'that'). When the two occurrences are found together, this can be made part of another hypothetical relationship. To construct a testable hypothesis users may need several 'if' and 'then' rules, each using codewords assigned to the data. The hypothesis test searches for cases where the particular codewords occur in the particular combinations the proposition requires. If these are present that case supports the proposition.

Clearly such procedures take us some way from working with the data itself. Users have to assume that the data supporting these hypothetical relationships is comparable and that coding has been done in a commensurate fashion, because they are working 'one level up', with relationships between the categories/codes. Yet the comparability of qualitative data is problematic. Rapport with respondents may vary, respondents may talk with different degrees of specificity, and so on. Hypothesis-testing will be discussed further, after considering another formal approach.

Work with codes is not the only approach to qualitative data analysis. There is a fundamental distinction between analysis based on *codes* and analysis based on *cases*. Case-oriented approaches, the basis of analytic induction, suit some analytic purposes. Analytic induction considers instances where some phenomenon of interest is present, on the basis that it occurs only when particular conditions apply. The idea is to isolate the essential conditions determining the phenomenon. In fact, analytic induction also needs to consider cases where the phenomenon is *not* present, to be certain there are no cases where the conditions postulated for the phenomenon to occur appear in the absence of the phenomenon (Hicks 1994 called this 'neo-analytic induction'). Neo-analytic induction is useful in analysing dynamic processes, especially where there is an interest in causality. The approach called Qualitative Comparative Analysis (Ragin 1987) thinks in terms of outcomes, comparing outcomes across multiple cases. Cases are literally recognized as such from the different outcomes they display. Cases are differentiated by the relationship between their outcomes and the components they share with other cases having different outcomes. Qualitative Comparative Analysis (QCA) uses logic

and mathematics to identify systematically 'universal conditions' which are always present in particular combinations when the phenomenon occurs. The 'QCA' software analyses conjunctures of particular elements of cases which bear on the outcome of a process.

We should note a further formal analytic approach using qualitative data. Event Structure Analysis represents series of events derived from fieldwork as logical structures, comprising elements and their connections. Like QCA it seeks explanatory models for interpreting sequences of events (Heise and Lewis 1988), having a chronological (process) orientation and concern with causal explanation. Its basic logic is that in each situation social actors have limited choices, and certain events cannot occur before their prerequisites. Based on these premises abstract logical structures of events are generated, and compared to actual event sequences. Event Structure Analysis is supported by a program called 'Ethno' (Heise 1991). The program enables users to model narrative accounts of event sequences and produce mathematically based causal accounts.

Like the hypothesis-testing approach of HyperRESEARCH, Qualitative Comparative Analysis and Event Structure Analysis involve high-level coding, but, unlike HyperRESEARCH, they also require extensive data reduction prior to analysis. Although the data that Qualitative Comparative Analysis and Event Structure Analysis use are qualitative, some do not regard these approaches as qualitative analysis at all. Their concern is that the analysis proceeds at a level too far removed from the original data and the context in which they were collected. They suggest that, if the data are going to be so stringently 'reduced', there is little point in collecting qualitatively 'rich' data in the first place. The status of these formal approaches is an important issue to which I will return.

There is a further formal approach that should be mentioned because it strongly embraces concerns about the importance of context. This approach is Formal Concept Analysis (FCA: Mackensen and Wille 1998) and, although it is mathematically based (on set theory), it supports the iterative process of formulating codes/categories familiar from grounded theory and which lies behind the code-and-retrieve packages. The prototype application is based on the TOSCANA data management system. FCA construes concepts as constituted by their 'extension' – all objects belonging to the concept – and 'intension' – all attributes valid for those objects. A concept can be graphically displayed in a line diagram or 'concept lattice' consisting of nodes, lines and labels. The extensions of the concept consist of all objects whose labels are attached to the node or to a node derivable by a descending path from the node, and its 'intensions' consist of all attributes whose names are attached to the node or to a node derivable by an ascending path from the node. The descending/ascending element reflects the hierarchy or 'levels' of data and concept. Analysis proceeds by traversing these paths from objects to attributes, investigating the concepts by 'zooming in' from a more abstract level to the data, or by following links from items of data to categories linked at several progressive levels of abstraction.

FCA offers a means to represent conceptual systems graphically while being able to return at any time to the data. The representation alerts users to concepts which are thinly supported (have few links to data), to empirical examples which contradict elements of postulated concepts, and to dependencies among categories. For example, research on the emergence of the notion of 'simplicity' as a criterion in classical music drew on a set of published sources in which the term was used, and various facets of these sources, including their date of publication. The pattern of diffusion of the 'simplicity' idea could be traced and key sources identified (as nodes with many links), allowing inferences about the degree of authority contemporaries accorded each source. The FCA approach uses mathematical techniques of conceptual and logical scaling to produce its concept lattices, but is more in line with the premises of qualitative research than the other formal approaches.

The three-fold typology of qualitative software can be regarded as successive generations (Kelle 1996). The first generation were word-processor and database management systems, which supported techniques like cut-and-paste but did not exceed the most basic requirements of data management. Specialized code-and-retrieve programs developed in the early 1980s contributed the ability to manage unstructured textual data and sophisticated cut-and-paste, indexing and memoing techniques. Most importantly, the second generation made coding and retrieval processes transparent. The third generation, with the ability to search for co-occurring codes, and methods to construct complex networks linking categories, codes, memos and text segments, provided facilities to support theory building and hypothesis testing. Mangabeira (1995) identifies the distinctive element of this third generation as being the model-building capabilities of the software.

The generational perspective has an important implication. As development proceeds and packages become more fully featured, a measure of convergence occurs. Exchanges between users and developers motivated many changes, others were imposed, the main example being the Windows operating system. Consequently the typology becomes less rigid; for example, one might regard WinMAX, the Windows release of the first generation package 'MAX', as a code-based theory builder rather than a code-and-retrieve package.

When qualitative software first emerged, it was informed by prevailing approaches to analysis. The main analytic metaphor was code-and-retrieve, and few looked to software to inspire new approaches to analysis. However, packages increasingly support procedures which are new or impractical without the computer. We can no longer argue that the software is simply an aid to code-and-retrieve or that code-and-retrieve is the essence of qualitative analysis. The question is, have users' practices changed accordingly?

WILL QUALITATIVE SOFTWARE KILL OFF MANUAL METHODS?

Many qualitative researchers remain committed to manual methods and, for certain purposes, manual methods have advantages over software, for

example, where setting up the data in the required format would take a disproportionate time relative to the scope of the analysis. No one should feel guilty about not using software or pressured into adopting it. However, a major benefit of qualitative software is that it obliges researchers to be clearer about their reasoning, and enables the analytic process to be more transparent and therefore accountable. These characteristics will shift community standards of analytic quality over time, and those relying on manual methods may find themselves having to invest inordinate effort in the paper-based equivalents of procedures which are done in seconds using software.

It is important to repeat that simply using CAQDAS does not mean the whole analytic process takes place 'within' the software. It is true that prototype 'expert systems' have emerged, which, for example, prompt users to consider whether the data with which they have supported an inference is adequate. Such prompts are only as good as the criteria 'wired into' the program; expert system software has a potential role in teaching, but the prototypes encourage users to override program advice if they wish.

It is pedagogically desirable that those new to qualitative methods should have their initial experiences using manual approaches, to gain a firm grounding in procedures that they may later automate. This also avoids conflict between learning the rudiments of method and learning the procedures of software use; the consensus is that people should have a reasonable grasp of the foundations of qualitative research first (Allatt and Benson 1991; Fielding and Lee 1996).

CAQDAS IN THE RESEARCH ENVIRONMENT

Sales data for one of the longest established packages, The Ethnograph, suggests that early users featured in applied fields such as social research in medical settings rather than in discipline-based academic research. This may relate to the hope that CAQDAS can resolve time pressure, although such hopes appear to be false (Lee and Fielding 1995). Research suggests this pattern of adoption may also account for the typical mode of use, which tends to exploit data management rather than conceptualizing or analytic features (Fielding and Lee 1998).

One might assume that research on CAQDAS users would be dominated by reports of bugs, crashes, program architecture and so on – that is, technical and practical issues. However, the focus groups and individual interviews that Ray Lee and I conducted with about 60 CAQDAS users in Britain (Fielding and Lee 1998) revealed that the issues preoccupying users were not simply technical but matters of the 'research environment'. Tangled up with software use were fundamentals of the social context of research – sponsors' demands, deadline pressures, power relations in research teams and with research subjects, and so on. Accounting for respondents' experiences means exploring how 'software issues' impact on 'research environment issues', and vice versa.

Take, for example, the apparently banal matter of availability. Originally users found product information elusive and distribution amateur. Academics who sought advice from their university's computing centre almost invariably reported that staff had never heard of CAQDAS. Social science computing began and ended with statistics packages. Graduate students were particularly likely to be referred back to their own department but regardless of status the general response was that no one had requested such software before and resources were not available. The same story generally prevailed when requesters approached their own departments for help, except, in some cases, ignorance was compounded by hostility to the idea of 'mechanizing' the craft of qualitative analysis. We documented cases where users faced opposition to CAQDAS from senior project staff or doctoral supervisors.

Would-be users who finally got the software they wanted did not usually do so by comparative evaluation and discriminating choice, but because someone in their social network had details of a particular package. Computing centres remained unwilling to help with support and training, because staff would have to be found to evaluate and then support the new software. Training is, even now, largely ad hoc and patchily distributed; for example, in an exchange on the email list for Atlas/ti users a would-be user in Texas appealed for training, which elicited an offer if the individual could get to Santa Barbara. It is a long way from Texas to California. Thus, the seemingly straightforward matter of obtaining a package ready for use on one's own machine invokes issues concerning the status of qualitative research, computing centre resources, the organization of the software market, and training.

I noted that choice is often ad hoc. Yet packages vary and users need to choose carefully, taking account of not only cost and the availability of support but also the nature of the analytic project(s) for which the software is needed, the type of work they are likely to do in future (applied/academic, policy-related/conceptualizing, solo/team and so on), the balance between ease-of-use and availability of sophisticated features, and other factors (for full discussion of 'choice' issues, see Weitzman and Miles 1995). While applied researchers faced fewer problems justifying acquisition, and research sponsors appear to favour CAQDAS for reasons of legitimation, applied researchers face their own problems. Applied research features tight deadlines and has relatively straightforward analytic requirements. Users complained that data entry and setting-up occupied a disproportionate time relative to the analysis the sponsor wanted, and some felt they were under-utilizing the package.

Users expected three main things from CAQDAS: that it would save time, make analysis more thorough, and support analytic procedures that were impractical manually. However, we found no case where users reported CAQDAS saved on analysis time. This is probably not susceptible to a technological fix. Coding and analysis are (and should be) intrinsically time-consuming. Not only have developers not claimed that using CAQDAS is

quicker, several emphasize the need for thoroughness in the coding process and reflect this in program design (this is not to say procedures for applying codes should not be simple and quick). What about the expectation that analysis would be more thorough? Users gave abundant testimony to this. Computers do not forget or mislay things, they always do the same procedure the same way, and, if users need reminding of how they have carried out some operation, the software can often help retrace steps. Users were in no doubt that CAQDAS can manage voluminous data more effectively than can paper-based methods.

But evidence was mixed on the support for sophisticated conceptual work. I already noted that some applied researchers found the pace of their work denied them time for creative play with advanced features; sometimes they simply did not have time to code all the data, seriously limiting the kind of analytic work possible. The value of CAQDAS to them was as an electronic filing cabinet, though some appeared to ease their frustration by promising themselves they would use the more advanced analytic features in their own time at some future point. Academic users were, generally, enthusiastic about the conceptualizing features, but reported problems learning how to use them. Graduate students were most enthusiastic about the conceptualizing features, perhaps suggesting a more open mind and fewer preconceptions.

Thoughts on the use of CAQDAS in the analytic process prompted reflections beyond technical, computing issues. Users tended initially to declare affiliation to a particular analytic approach, most often 'grounded theory', citing the appeal of rigorous analysis without quantification, the elevation of 'closeness to the data', and conceptualization emergent from the data rather than from formal theory. However, when we invited users to connect the elements of grounded theory procedure to the support offered by particular packages, accounts were seldom based on direct adherence to grounded theory procedures. It seems that users did not expect CAQDAS to support each element of grounded theory; they wanted something more flexible than that. Nor is the software as closely informed by grounded theory as the developers' early claims suggested. It goes little further than that CAQDAS packages are based on coding data segments and writing analytic memos. Thus, asserted affiliation to grounded theory should not determine our view of how qualitative software supports qualitative analysis. To suggest that CAQDAS particularly supports grounded theory work is as plausible as suggesting that all qualitative research is grounded theory. Indeed research can be both rigorous and illuminating without using a 'standard' approach at all (Weaver and Atkinson 1994).

Recent versions of qualitative software offer features which enable kinds of work which were not previously supported. It is particularly intriguing whether these features are being exploited, in light of the limited use of the more sophisticated analytic features noted in our research. Similarly, the extent of use of QCA and FCA would be of great interest, as these are associated with more formal approaches to qualitative data analysis, which are

of considerable interest to researchers well outside the qualitative camp (Lindenberg 1998).

NEW TECHNOLOGIES = NEW ANALYSES?

While developers foresee an increasing integration of software tools this is unlikely to result in a 'superpackage' having every feature offered by any package. Instead, users will increasingly be able to transfer projects between packages, with coding done in one package, conceptual mapping in another, and so on. To enable this, there will be some standardization of procedures, a trend encouraged by operation under Windows. There is also a tendency to more visually based analytic and data presentation strategies. Current examples include the tree diagrams of NUD*IST4, N-vivo's visual approach to modelling, and the 'network views' of Atlas/ti. The latter also supports coding of graphic images. Packages also increasingly provide facilities to export worked projects to websites for joint work and dissemination. Other changes include more automated coding, the general move to provide full suites of Boolean operators, and better-designed interfaces with increased use of 'drag and drop' functionality. Several developers are adding formatted text features, to preserve the cues to context these can offer, allow code assignments to stay linked to text as projects are moved between packages, and accommodate accents and non-standard symbols.

Adjacent technological developments also have implications for qualitative research. Data transcription is laborious, prone to error, expensive and subject to the sensibilities of transcriptionists. Voice recognition software, which converts speech to text, is improving. Software like DragonDictate, VoiceType, and Kurzweil Voice still pose considerable requirements in training the recognition facility to the user's voice and checking output for accuracy, nor are they cheap. But improvements are continual and prices should be set against the cost of transcription services. An alternative is 'direct transcription software', where speech is recorded on a CD-ROM and codes are applied not to text but to the sound segments themselves (e.g. *Code-A-Text, C-I-SAID, Qualitative Media Analyzer*). During retrieval the researcher can listen to the actual data as well as read them as text. Pitch, modulation and other sound features can be monitored, allowing inferences from tone of voice as well as the words uttered. It is worth noting that the developer of *Code-A-Text* uses it in analysis of clinical/therapeutic interviews and in conversation analysis. Such software may also be useful in market research, where time does not permit transcription, or in making a first high-level pass through the data to identify broad themes.

While technological developments contribute to change in approaches to analysis, they are but one influence among many. The new technologies available to qualitative researchers cannot but make for innovation and experiment (and an accompanying degree of confusion), but their transformative potential (Lee 1995) is not so strong as to change the external forces

which shape the qualitative research world or the internal variety which makes qualitative work so interestingly diverse. However, if we narrow our attention from the macro level, qualitative software and other IT, notably the Internet, does allow those who are so inclined to further develop the progress qualitative research shows toward greater formalization, and to help those who are so inclined to seek new community standards of participation in the products of qualitative research (see Mann and Stewart 2000). These themes are pursued below.

THE POTENTIAL TO GENERALIZE AND THE INCLINATION TO PARTICULARIZE

Critics of qualitative research cite its lack of formality and cumulativeness. Until recently, this has been amplified by inattention to analytic principles and procedures in the qualitative methods literature. These traits compare unfavourably with the formal and systematic character of statistical analysis and survey methods. While analysing words is different from analysing numbers this is not a warrant to be evasive or mystical about our analytic procedures. Both formalism, the drive to construct algebraic expressions of axiomatic knowledge, and formalization, the effort to codify methodological procedures, are relevant here. As statistics has developed new tools, it has prompted more stringent criteria and procedures of quantitative analysis. In qualitative research, formalizing influences have concentrated on principles for data collection. The network of developers, users and methodologists associated with qualitative software has provided an impetus to formalization of analytic procedure. Further, some methodologists have sought to make the object of qualitative analysis the identification of formal conditions which capture, for example, cause and effect relationships. One might cite work by mathematical sociologists in modelling behaviour using ethnographic data (Abell 1988); such an orientation influenced the work in Event Structure Analysis (Heise 1991). Technology increasingly facilitates that kind of analysis. Yet, perhaps because such work is more keenly attended-to outside qualitative research (for example, in rational choice theory and social simulation), it attracts less notice than the more vocal analytic orientation which celebrates relativism and repudiates the search for formal knowledge and generalization.

The point about these ruptures relevant to qualitative software is that, while these commitments colour our attitudes to new technology, they have not come from technological developments but from adherence to intellectual traditions which have grown up in the theory and practice of qualitative research. The relativist and postmodern positions are products of the Romantic tradition in philosophy (Strong and Dingwall 1997), and one impetus to 'emotionalism' came from applied studies in medical settings. The new research technologies are not the source of new analyses, though they may be their vehicle. We might take for example the 'QCA' software

developed by Drass (1989) and the hypertext-linking approach Weaver and Atkinson (1994) document in their writings on the use of GUIDE. The former addresses causal analysis, the latter celebrates chance discovery and the multiple meanings any text can support. The QCA approach is directed to formalism and a positivist understanding of data as facts that can be abstracted. Weaver and Atkinson (1994) pursue an ever-changing, suggestive analysis and a postmodern conception of data as infinitely contested interpretations. These approaches share no ground. Yet both have been operationalized by qualitative software in a way that would not otherwise be possible.

In fact, self-conscious 'landmark' texts, like Denzin and Lincoln's (2001) handbook, make it apparent that the growth and subdivision of schools of analytic thought has proceeded quite independently of IT developments. If we accept Denzin and Lincoln's classifications, we have several postmodern approaches to analysis, along with post-positivism and the evergreen interactionism. The liberal use of the prefix 'post' suggests that these are rebrandings of familiar branches of qualitative analysis. It is most unlikely that qualitative software will, in itself, prompt the invention of new methods or schools of thought. So far, at least, where technology transforms social science, it transforms procedure. An interview administered by Netcam allows a researcher in Guildford to do 'fieldwork' in Melbourne but the researcher will still have to wrestle with rapport, or its lack, when interpreting the data. There will be a difference of degree but not of kind.

The kudos associated with discovery of the 'new' encourages authors, editors and publishers readily to assign self-labelled new approaches to a new category. Notice, for instance, that, like the Denzin and Lincoln (2001) handbook, the volume you are reading has one, standalone chapter on 'software'. Why, one wonders, are software implications not discussed in relation to each of the fieldwork topics and analytic procedures? Instead, software is depicted as 'an approach' in itself (in the original Denzin and Lincoln volume (1994), it is a 'method of analysis': see Table 1.1, p. 12). No wonder those unacquainted with CAQDAS think it is a new and technicist approach, to put alongside critical theory, feminism and the rest.

It should therefore be said that there is little at the level of epistemology or approach to analysis in qualitative software which is not derivative of thinking that existed before a mouse was anything but an animal that squeaks. This is not to say that CAQDAS does nothing new. It allows us to operationalize procedures and approaches to analysis whose logical possibility was identified but whose demands were entirely impractical before the computer. An example might be the 'system closure' concept in NUD*IST4 and N-vivo (for a good practical account of using the latter, see Bryman 2001). Any 'transformation' that emerges will come as a result of being able now to test the consequences of procedures like QCA which, if its proponents are right, may enable credible claims to derive causal explanation inductively from non-numerical data. That would undeniably be a big thing, even a transformation, but it would not represent a new logic. We still confront the obstacles that were apparent to Weber. New technologies even mean we

have to negotiate new ramifications of those problems; for instance, the Netcam example suggests that, in assessing context effects, we will have to make inferences about the effects of not being physically co-present. Software will enable us to tag and recover instances where those effects intrude on our analysis, though, and to retrace our analytic steps using 'audit trails', which may help us weigh up their impact. With technology we can do more, but we also have more to do.

FACILITATING TEAMWORK, ENABLING DEBATE, ENCOURAGING REFLEXIVITY

It takes time for markets to understand technologies, and what may eventually be prevalent uses may not be the obvious or the intended (e.g. the telephone was originally envisaged as a broadcast medium). One non-obvious use of qualitative software that could be very significant is in facilitating working cooperatively. CAQDAS packages offer a number of features to support joint work. In some cases, such as that of FCA, the idea of collaborative interrogation of the data is the essence of the software design. A virtue of team research is that it forces individuals to be more explicit about their reasoning. Software which enables individual inputs to be traced via 'audit trails' and work to be exported to websites for comment by dispersed team members can encourage sharper thinking and more clarity about how and why a particular analytic decision has been made. Where such features are used to encourage participation in analysis by interested outsiders they can enable debate which enhances the analysis and brings the findings to wider audiences. If the research subjects are encouraged to use these means to participate in the stages of inquiry subsequent to data collection we might even see a change in community understandings of research and what it is for. There is a particular potential for hypertext-based projects here, where outsiders might use the technology to construct their own interpretations of the data (Weaver and Atkinson 1994). The strong recent emphasis on relativism has given matters to do with reflexivity extra prominence; we increasingly recognize the need to track the effects of our own subjectivity and bring it to bear in our analysis. Again, CAQDAS can play a part, because it makes it easy to review the effects of sequences of program operations and to recast elements of the analysis to gauge their effects.

Qualitative software appears to have contributed to the rehabilitation and wider use of qualitative methods, by affording the technical appearance of formality that computer procedures have long lent to statistical research (Agar 1991). This helps establish credibility with research sponsors, be they government funding agencies, public service organizations or the private sector. Like the date with an old beau, qualitative research has also enjoyed new relationships on the rebound, as these selfsame funding bodies recognize the failures and limits of quantitative research. The analogy contains a warning. New enthusiasms are volatile. Qualitative research may enjoy a

new vogue but it is conditional. The old prejudices remain. If the practice of qualitative research confirms that it is akin to writing fiction or dogma, betrays an inability to decide what the data mean, or the techniques are 'non-transferable' (cannot be taught), qualitative research will be back out in the cold. Technology does not obscure sloppy thinking, it exposes it.

These views are, of course, informed by a particular view of what research is for. Research produces knowledge. It may be knowledge for-its-own-sake or it may be applied, but our audience is the research community, including those who use research to make policy decisions which affect the community. In this context, both the appearance and the reality of formalization, of which research technology is a part, are important features of the qualitative scene. We can contrast this with another view, where we seek new audiences by moving away from conventional forms of research writing (Richardson 1994). This approach also often sees qualitative research as a means of empowerment of research subjects.

But regardless of our conception of research, qualitative software can change our relation to our audiences. The transparency of the analytic process, where software allows users to recover and display the steps in their reasoning, opens our work to discussion with others. For example, the user experiences research documented cases where these features encouraged applied research sponsors to play a part in the analysis. The ability to export parts of our data, or our write-up, to a website means that others can play a part. At minimum, research subjects and other participants are enabled to comment on the data and/or the analysis. This may make inquiry more open-ended, give voice to those whose account is normally silenced, and make researchers more accountable. Such possibilities do seem to address the business of 'empowerment'.

AUTOMATING THE INEFFABLE?

Qualitative software has contributed greatly to the research community. For methodologists, its real significance does not lie in technology or technique but in exposing areas in which accustomed reasoning is muddy, and enabling experiment with approaches which looked promising but whose resource requirements exceeded the capacities of manual methods. CAQDAS has stimulated a reappraisal of qualitative methodology. It has prompted us to reassess accepted epistemological canons (such as the role of hypothesis-testing in qualitative work), provoked new methodological debates (such as those over the status of coding or grounded theory) and tested new analytic approaches (such as qualitative comparative analysis or using hypertext as a freeform method of discovery). For researchers its benefits include data management capacities unquestionably superior to paper-based systems, facilitation of the orderly and accountable practice of analysis, and features which invite us to extract the maximum from our data. For research audiences, qualitative software augments the claim of

qualitative research to legitimacy, but, more interestingly, beckons them to get more involved in the research rather than regarding it as a product they simply consume.

There are demerits too. For example, early users endured endless bugs, crashes and quirks, and even now the chance to make an informed choice and/or to gain access to systematic training is available to few. Opening up our texts via hypertext-based projects allows untutored use, which could result in serious disputes with research subjects and the misrepresentation of project findings. Automating code assignment allows blanket recoding rather than careful inspection of each segment before a code is assigned. Program complexity means that sometimes users have only a vague idea of what particular operations have actually done. Neo-quantification of program output may encourage inappropriate but apparently precise numerical analyses. These and other demerits may be regarded as elements of the one great problem, that qualitative researchers will claim for their findings, analyses and interpretations more than they should. So what's new? The field will respond to such problems as it always has, not by embracing articulated standardized criteria of validity, but by critical peer review, liberally interpreted community standards of analytic adequacy, and learning from experience (Altheide and Johnson 1994).

Qualitative research survived its years as the poor cousin of quantitative social science by creativity. It was 'a method of discovery'. It had a place even in survey research as a method to use at the pilot stage, because it was a source of ideas, hunches and hypotheses. Those who use qualitative software testify both that they get ideas from working with the software and that they get ideas in the traditional ways, in the bath, in the middle of a conversation, while landing at night. Using software won't prevent the 'eureka' effect but it won't guarantee it either.

REFERENCES

Abell, P. (1988) 'The "structuration" of action: inference and comparative narratives', in N. Fielding (ed.) *Actions and Structure.* London: Sage.

Agar, M. (1991) 'The right brain strikes back', in N. Fielding and R. Lee (eds) *Using Computers in Qualitative Research*. London: Sage.

Allatt, P. and Benson, L. (1991) 'Computing and qualitative analysis: issues in research methods teaching', in N. Fielding and R. Lee (eds) *Using Computers in Qualitative Research*. London: Sage.

Altheide, D. and Johnson, J. (1994) 'Criteria for assessing interpretive validity in qualitative research', in N. Denzin and Y. Lincoln (eds) *Handbook of Qualitative Research*, 1st edn. London: Sage.

Berelson, B. (1952) *Content Analysis in Communications Research.* New York: Free Press.

Bryman, A. (2001) *Social Research Methods*. Oxford: Oxford University Press.

Denzin, N. and Lincoln, Y. (eds) (2001) *Handbook of Qualitative Research*, 1st edn. Thousand Oaks, CA: Sage.

Drass, K. (1989) 'Text analysis and text analysis software: a comparison of assumptions', in G. Blank *et al*. (eds) *New Technology in Sociology: Practical Applications in Research and Work.* New Brunswick, NJ: Transaction.

Fielding, N. and Lee, R. (1996) 'Diffusion of a methodological innovation: computer-assisted qualitative data analysis in the UK', *Current Sociology* 44(3): 242–258.

Fielding, N. and Lee, R. (1998) *Computer Analysis and Qualitative Research*. London: Sage.

Heise, D. (1991) 'Event structure analysis: a qualitative model of quantitative research', in N. Fielding and R. Lee (eds) *Using Computers in Qualitative Research*. London: Sage.

Heise, D. and Lewis, E. (1988) *Introduction to ETHNO: Version 2*. Raleigh, NC: National Collegiate Software Clearinghouse.

Hicks, A. (1994) 'Qualitative comparative analysis and analytic induction: the case for the emergence of the social security state', *Sociological Methods and Research* 23: 86–113.

Kelle, U. (1996) 'Computer-aided qualitative data analysis: an overview', in C. Zull, J. Harkness and J. Hoffmeyer-Zlotnik (eds) *Text Analysis and Computers*. Mannheim: ZUMA.

Kelle, U. (1997) 'Theory building in qualitative research and computer programs for the management of textual data', *Sociological Research Online* 2(2).

Lee, R. (ed.) (1995) *Information Technology for the Social Scientist*. London: UCL Press.

Lee, R. and Fielding, N. (1991) 'Computing for qualitative research: options, problems and potential', in N. Fielding and R. Lee (eds) *Using Computers in Qualitative Research.* London: Sage.

Lee, R. and Fielding, N. (1995) 'Confronting CAQDAS: choice and contingency', in R. Burgess (ed.) *Studies in Qualitative Methodology*. New York: JAI Press.

Lindenberg, S. (1998) 'The influence of simplification on explananda: phenomenon-centred versus choice-centred theories in the social sciences', in H-P. Blossfeld and G. Prein (eds) *Rational Choice Theory and Large-scale Data Analysis*. Boulder, CO: Westview.

Mackensen, K. and Wille, U. (1998) 'Qualitative text analysis supported by conceptual data system', unpublished working paper, Mannheim: ZUMA.

Mangabeira, W. (1995) 'Computer assistance, qualitative analysis and model building', in R.M. Lee (ed.) *Information Technology for the Social Scientist*. London: UCL Press.

Mann, C. and Stewart, F. (2000) *Internet Communication and Qualitative Research: A Handbook for Researching Online.* London: Sage.

Ragin, C. (1987) *The Comparative Method: Beyond Qualitative and Quantitative Strategies.* Berkeley, CA: University of California Press.

Richardson, L. (1994) 'Writing: a method of inquiry', in N. Denzin and Y. Lincoln (eds) *Handbook of Qualitative Research*, 1st edn. London: Sage.

Strong, P. and Dingwall, R. (1997) 'Romantics and Stoics', in G. Miller and R. Dingwall (eds) *Context and Method in Qualitative Research*. London: Sage.

Weaver, A. and Atkinson, P. (1994) *Microcomputing and Qualitative Data Analysis*. Aldershot: Avebury.

Weitzman, E. and Miles, M. (1995) *Computer Programs for Qualitative Data Analysis*. Thousand Oaks, CA: Sage.