RESEARCH · QUALITATIVE RESEARCH

Coding Qualitative Data: Manual and Software-Assisted Approaches

How to code qualitative data, covering what a code is, first and second cycle coding methods, building a codebook, coding in teams and checking agreement, manual methods, qualitative data analysis software, computational and AI-assisted coding, and common mistakes.

ModQu research article · 2,005 words · 20 references (APA 7th edition)

Coding is the core activity of most qualitative analysis. It means reading through interview transcripts, field notes, open survey answers or documents, and attaching short labels, called **codes**, to segments of text that are relevant to the research question. Codes allow the analyst to gather similar material together, compare it across participants and build it into themes, categories or theories. Saldaña (2016) describes a code as a word or short phrase that symbolically assigns a summative, salient, essence-capturing or evocative attribute to a portion of data. This article explains how coding works, the main coding methods, how to code in teams, and how manual coding compares with software and newer computational tools.

What coding does

Coding serves several purposes at once. It **organises** large amounts of text so that material on the same topic can be retrieved. It **condenses** the data, reducing hours of conversation to a manageable set of ideas. And it is an act of **interpretation**: deciding what a passage is about, and what to call it, already involves analytic judgement. Miles et al. (2014) describe coding as a form of deep reflection about the data, not a clerical task that comes before analysis.

Codes can be **descriptive**, summarising the topic of a passage ("price"), or **interpretive**, capturing what the passage means ("price as a signal of quality"). Early in analysis, descriptive codes are common; as analysis proceeds, codes become more interpretive and more closely tied to the research question.

First and second cycle coding

Saldaña (2016) distinguishes **first cycle** methods, applied to the raw data, from **second cycle** methods, which work with the codes produced in the first cycle to identify patterns, categories and themes.

Six ways to code one passage

"I'd been meaning to switch for ages, but it always felt like a hassle..."

Method	Cycle	Example code
Descriptive	First	Switching supplier
In vivo	First	"Felt like a hassle"
Process	First	Putting off switching; acting on a neighbour's example
Values	First	Valuing ease over savings
Emotion	First	Frustration; resolve
Pattern	Second	Social proof as a trigger for overdue action

Based on Saldaña (2016); examples by ModQu ModQu.com

Figure 1. Common coding methods, applied to one hypothetical passage about switching energy supplier. Based on Saldaña (2016); examples by ModQu.

Common first cycle methods include:

- **Descriptive coding**: summarising the topic of a passage in a word or short phrase.
- **In vivo coding**: using the participant's own words as the code, which keeps the analysis close to their language.
- **Process coding**: using verb forms ending in "-ing" to capture actions and processes, such as "checking reviews" or "delaying the decision". Charmaz (2014) recommends this approach in grounded theory because it keeps attention on what people do.
- **Values coding**: capturing participants' values, attitudes and beliefs.
- **Emotion coding**: labelling emotions expressed or described.

Second cycle methods include **pattern** or **focused coding**, which groups first cycle codes into a smaller number of categories, and **theoretical coding**, which relates categories to each other to build an explanation. In thematic analysis, the second cycle is the stage at which codes are clustered into candidate themes (Braun & Clarke, 2006).

Inductive, deductive and hybrid coding

Codes can be developed from the data, **inductively**, or drawn from existing theory or the research questions, **deductively**. Hsieh and Shannon (2005) described these as conventional and directed approaches to qualitative content analysis. Many studies use a **hybrid** approach: a starting list of codes drawn from theory or the brief, applied to the data and extended with new codes when material does not fit (Fereday & Muir-Cochrane, 2006).

Inductive coding is more open to surprises; deductive coding is quicker and makes it easier to compare results with other studies. The right balance depends on how much is already known about the topic and what the study needs to deliver.

Building a codebook

A **codebook** lists each code with a definition, guidance on when to use it and when not to, and an example. MacQueen et al. (1998) recommended that codebooks for team-based analysis include, for each code, a short label, a full definition, inclusion and exclusion criteria, and typical and atypical examples. A good codebook makes coding consistent across the dataset and across coders, and it records how the meaning of codes was decided.

Codebooks evolve. As coding proceeds, codes are added, merged, split and redefined. Recording these changes, with the reasons for them, is part of the audit trail that supports the dependability of the findings.

Coding in teams

When several people code the same data, they need to apply codes consistently. A typical process is for two or more coders to code the same sample of transcripts independently, compare their coding, discuss differences and refine the codebook, and repeat until they apply codes in a similar way. The remaining data are then divided among coders, with periodic checks.

Agreement between coders can be measured. The simplest measure is the percentage of segments on which coders agree. Because some agreement happens by chance, statistics such as **Cohen's kappa** adjust for chance agreement (Cohen, 1960). Hruschka et al. (2004) described their experience of measuring reliability in a large team study and found that achieving acceptable agreement required several rounds of codebook revision. Campbell et al. (2013) showed that agreement depends heavily on how text is divided into units before coding, and proposed procedures for dealing with this.

Not all qualitative researchers accept that agreement should be measured. O'Connor and Joffe (2020) reviewed the debate and concluded that inter-coder reliability can be valuable when a codebook will be applied by a team, or when findings need to persuade audiences familiar with quantitative standards, but that it does not suit approaches in which interpretation is expected to differ between analysts, such as reflexive thematic analysis (Braun & Clarke, 2021).

Manual coding

Coding can be done entirely by hand. Analysts print transcripts, write codes in the margins, highlight passages in different colours, and cut out extracts to sort into piles. Spreadsheets and word processors can serve the same purpose on screen: each row a segment of text, each column a code or theme. The **framework method** organises coded data into a matrix with one row per participant and one column per code, summarising each participant's view in each cell (Gale et al., 2013).

Manual coding keeps the analyst close to the data and is practical for small studies. It becomes cumbersome with large datasets, many codes or several coders, when retrieving all passages with a given code becomes slow.

Qualitative data analysis software

Computer-assisted qualitative data analysis software (often called CAQDAS or QDAS), such as NVivo, ATLAS.ti and MAXQDA, supports coding by storing documents, applying codes to selected text, retrieving all text with a given code, and searching for co-occurrences of codes. Many packages also handle audio, video, images and survey data, and support memos and visual maps of codes.

Fielding and Lee (1998) studied early users of such software and found that it made analysis more systematic and transparent, but that some researchers worried it would encourage mechanical coding at the expense of interpretation. Welsh (2002) found that software was most useful for managing data and searching, while interpretation still depended on the analyst's close reading. Silver and Lewins (2014) and Bazeley and Jackson (2013) provide practical guides to using software well, emphasising that the analyst should decide on the analytic approach first and then use the software to support it.

Woods et al. (2016) reviewed published studies that used NVivo or ATLAS.ti over two decades and found that most used the software mainly for coding and retrieval, making little use of its more advanced features. The software does not perform the analysis; it makes certain tasks faster and the process easier to document.

Computational and AI-assisted coding

Computational methods can analyse far larger volumes of text than any human team. Grimmer and Stewart (2013) reviewed automated methods for analysing political texts, such as dictionary methods, supervised classification and topic models, and concluded that they can be powerful but that all such methods are "wrong" in the sense that they simplify language, so their results must be validated against careful human reading. Nelson (2020) proposed **computational grounded theory**, in which computational methods first identify patterns in a large corpus, the researcher then reads closely to interpret them, and the patterns are finally checked again across the corpus.

Large language models can now suggest codes, summarise passages and apply a codebook to new text. They can speed up the first pass through a large dataset and help an analyst check whether a code has been applied consistently. They also make errors, may miss subtle or unusual meanings and can reproduce the assumptions in the instructions they are given. Their output should be treated like the work of an untrained assistant: useful as a starting point, but checked against the data by a human analyst who remains responsible for the interpretation. Studies that use such tools should report how they were used and how their output was checked.

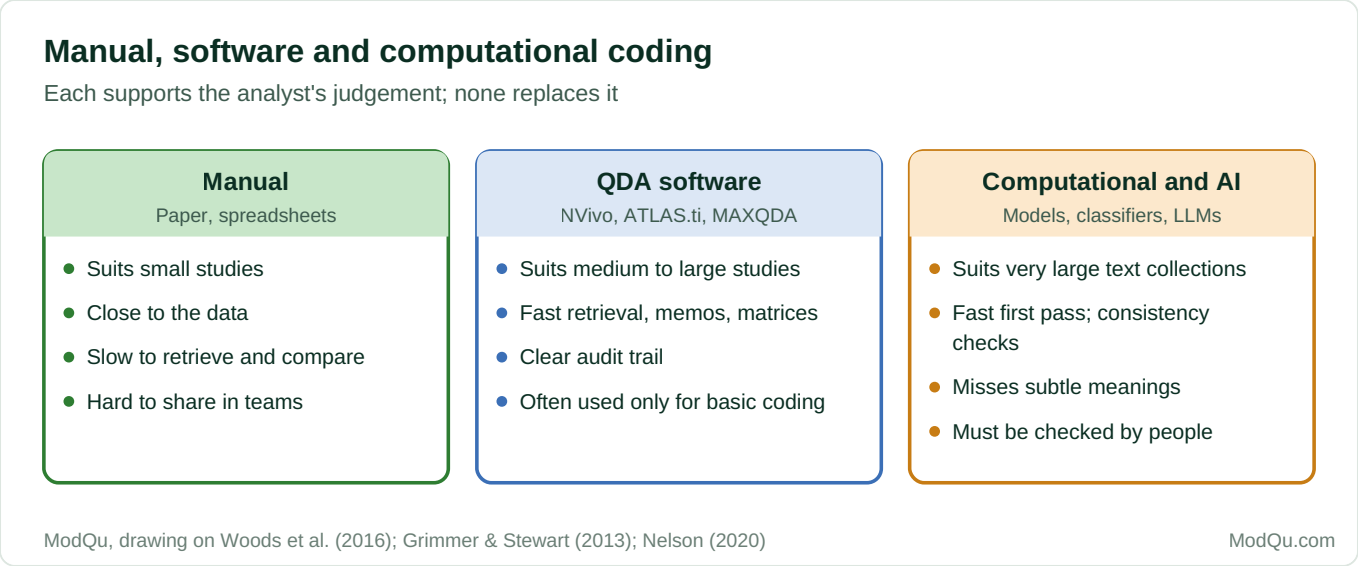


Figure 2. Three ways to support coding, with their strengths and cautions. Diagram by ModQu, drawing on Woods et al. (2016), Grimmer and Stewart (2013) and Nelson (2020).

Memos alongside codes

Codes on their own lose the reasoning behind them. **Analytic memos** are short notes written during coding that record why a code was created, what it seems to mean, how it relates to other codes and what questions it raises. Saldaña (2016) recommends writing memos frequently, treating them as the place where analysis happens, and Charmaz (2014) treats memo writing as the bridge between coding and the written account. A memo might note that a code first applied to complaints about delivery is starting to appear in accounts of praise as well, suggesting that the underlying issue is predictability rather than speed.

Memos also support transparency. When codes are merged or redefined, a memo records the decision and the reasons for it, so that others can follow how the analysis developed.

Practical tips

A few practical habits make coding more reliable and less laborious:

- **Code a small sample first.** Code three or four transcripts in full, then pause to review the codes, merge duplicates and write definitions before coding the rest.
- **Keep segments meaningful.** Code passages long enough to keep their context, usually a few sentences, rather than single words.
- **Allow multiple codes.** A single passage often says several things; apply every relevant code.
- **Return to earlier data.** When a new code is created late in the process, go back to earlier transcripts to see whether it applies there too.
- **Track coverage.** Check that every transcript has been coded with equal care, and that no participant's account has been left largely uncoded.

Common mistakes

Several mistakes recur in coding:

- **Coding too little.** Coding only passages that support an expected finding, and ignoring the rest.
- **Too many codes.** Hundreds of narrow codes that are never consolidated into categories, leaving the analyst lost in detail.
- **Codes that are only topics.** Coding by interview question ("question 3: views on price") rather than by meaning.
- **Undefined codes.** Codes without definitions, applied inconsistently as the analyst's understanding drifts.
- **Coding without memos.** Failing to record why codes were created and how they relate, which loses the analytic thinking behind the codes.

A brief example

Consider a passage from a hypothetical interview with a customer who switched energy supplier: "I'd been meaning to switch for ages, but it always felt like a hassle. Then my neighbour said she'd done it in ten minutes, and I thought, right, I'm doing it tonight." A descriptive code might be "switching". An in vivo code might be "felt like a hassle". Process codes might be "putting off switching" and "acting on a neighbour's example". In a second cycle, these codes might join others under a category such as "social proof as a trigger for overdue action", which links this passage to similar accounts across the dataset.

Conclusion

Coding turns raw qualitative data into material that can be analysed. It is both organisation and interpretation. Good coding uses methods suited to the research question, develops and documents codes carefully, applies them consistently, and moves from first cycle labels to categories and themes that answer the question. Software and computational tools can make coding faster and more transparent, but they support the analyst's judgement rather than replacing it.

References

Bazeley, P., & Jackson, K. (2013). *Qualitative data analysis with NVivo* (2nd ed.). SAGE.

Braun, V., & Clarke, V. (2006). Using thematic analysis in psychology. *Qualitative Research in Psychology, 3*(2), 77–101. https://doi.org/10.1191/1478088706qp063oa

Braun, V., & Clarke, V. (2021). One size fits all? What counts as quality practice in (reflexive) thematic analysis? *Qualitative Research in Psychology, 18*(3), 328–352. https://doi.org/10.1080/14780887.2020.1769238

Campbell, J. L., Quincy, C., Osserman, J., & Pedersen, O. K. (2013). Coding in-depth semistructured interviews: Problems of unitization and intercoder reliability and agreement. *Sociological Methods & Research, 42*(3), 294–320. https://doi.org/10.1177/0049124113500475

Charmaz, K. (2014). *Constructing grounded theory* (2nd ed.). SAGE.

Cohen, J. (1960). A coefficient of agreement for nominal scales. *Educational and Psychological Measurement, 20*(1), 37–46. https://doi.org/10.1177/001316446002000104

Fereday, J., & Muir-Cochrane, E. (2006). Demonstrating rigor using thematic analysis: A hybrid approach of inductive and deductive coding and theme development. *International Journal of Qualitative Methods, 5*(1), 80–92. https://doi.org/10.1177/160940690600500107

Fielding, N. G., & Lee, R. M. (1998). *Computer analysis and qualitative research*. SAGE.

Gale, N. K., Heath, G., Cameron, E., Rashid, S., & Redwood, S. (2013). Using the framework method for the analysis of qualitative data in multi-disciplinary health research. *BMC Medical Research Methodology, 13*, Article 117. https://doi.org/10.1186/1471-2288-13-117

Grimmer, J., & Stewart, B. M. (2013). Text as data: The promise and pitfalls of automatic content analysis methods for political texts. *Political Analysis, 21*(3), 267–297. https://doi.org/10.1093/pan/mps028

Hruschka, D. J., Schwartz, D., St. John, D. C., Picone-Decaro, E., Jenkins, R. A., & Carey, J. W. (2004). Reliability in coding open-ended data: Lessons learned from HIV behavioral research. *Field Methods, 16*(3), 307–331. https://doi.org/10.1177/1525822X04266540

Hsieh, H.-F., & Shannon, S. E. (2005). Three approaches to qualitative content analysis. *Qualitative Health Research, 15*(9), 1277–1288. https://doi.org/10.1177/1049732305276687

MacQueen, K. M., McLellan, E., Kay, K., & Milstein, B. (1998). Codebook development for team-based qualitative analysis. *Cultural Anthropology Methods, 10*(2), 31–36. https://doi.org/10.1177/1525822X980100020301

Miles, M. B., Huberman, A. M., & Saldaña, J. (2014). *Qualitative data analysis: A methods sourcebook* (3rd ed.). SAGE.

Nelson, L. K. (2020). Computational grounded theory: A methodological framework. *Sociological Methods & Research, 49*(1), 3–42. https://doi.org/10.1177/0049124117729703

O'Connor, C., & Joffe, H. (2020). Intercoder reliability in qualitative research: Debates and practical guidelines. *International Journal of Qualitative Methods, 19*. https://doi.org/10.1177/1609406919899220

Saldaña, J. (2016). *The coding manual for qualitative researchers* (3rd ed.). SAGE.

Silver, C., & Lewins, A. (2014). *Using software in qualitative research: A step-by-step guide* (2nd ed.). SAGE.

Welsh, E. (2002). Dealing with data: Using NVivo in the qualitative data analysis process. *Forum Qualitative Sozialforschung / Forum: Qualitative Social Research, 3*(2), Article 26. https://doi.org/10.17169/fqs-3.2.865

Woods, M., Paulus, T., Atkins, D. P., & Macklin, R. (2016). Advancing qualitative research using qualitative data analysis software (QDAS)? Reviewing potential versus practice in published studies using ATLAS.ti and NVivo, 1994–2013. *Social Science Computer Review, 34*(5), 597–617. https://doi.org/10.1177/0894439315596311