User's Manual

Contributions are more than welcome.

0. Design principles

Reliability: This is the most important thing. In order to maximize the reliability, every change will write to the *.rqda file directly, so you don't have to manually save your project.

File management capabilities: This package is written for my own research, aiming at analyzing newspaper reports. In this case, there are lots of files, and I have put much effect on the management and organization of files (Sorting, Show coded files only, Show uncoded files only, Show those matching a search criterion only etc.).

Mixed methods study: RQDA aims to support mixed methods studies. One example is the Cases Tab, which enables your to perform a Qualitative Comparative Study (see [QCA3](#) or [QCA](#) for more).

1. Project

You can create a new project in by clicking the "New Project" button in Project Tab. RQDA will create a *.rqda file, which is a SQLite database. All the information (e.g. files, code list, all kinds of memo, coding, and relation between codes or between files etc.) is stored in that single file. A single *.rqda file makes backup or data migration much easier.

If you already have a project, you can open it by clicking the "Open Project" button. Only when a project is open can you conduct futher analysis. You can click on the "Project" tab to inspect which *.rqda file you are working on.

You can back up your project by clicking the "Backup Project" button. For example, I am working on demo.rqda file, and click the "Backup Project" button, I obtain a backup file named demo.rqda_17245808122008. The suffix means that the backup action happened at 17:24:58 of 8 Dec, 2008.

Before you go on, you may would like to set the "Name of coder" in the "Settings" tab, so you can find out who is responsible for this RQDA project. You can replace the default with your name and then click the "OK" button.

2. Files

After opening a project, you can import files by clicking the "Import" button in the "Files" tab. By default, the file should be a plain text with an ASCII encoding, otherwise, you should set the "File Encoding" in "Settings" tab before you import the files.

If you want to import a lots of files, you can use the R function write.FileList() to import them. For more information about its usage, please consult ?write.FileList. In addition, from the pop-up menu in the "File" tab you can add new file directly by typing or pasting content: right-click - Add new file... - type new content or paste content - click "save to project" - input the file name and click "Ok". After that, you can close the "add new file" widget.

You can delete files. First select the file you want to delete, then click the "Delete" button in the "Files" tab. Similarly, you can rename the selected file (one file at the time). You can select a file, and click the "Open" button to open the file, so you can read its content. You can open a file by double clicking as well.

You can show only a subset of files by using the context menu that appears by right-clicking on a file. For example, you can show the uncoded files only, or the coded files only. After that, if you want to show all the files again, you can select "Show All Sorted By Imported Time". You can also show a subset of files matching the a search criterion by selecting "Search files ..." from the context menu. For more about the setting of the search criterion, you can refer to the help page by entering ?SearchFiles at the R prompt.

You can sort the files by file names as well. See tip 3 for more information.

You can calculate the total number of files in a project by using the R command length(GetFileId()), the total number of uncoded file by using the command length(GetFileId(type="uncoded")), and the number of coded files by using the command length(GetFileId(type="coded")).

3. Codes

Basically, the essence of QDA is to apply a label to a text segment. The label is usually a meaningful concept in the research, which is called **"code"** here. Thus you have to generate a code list. You can add a code to the list by clicking the "Add" button in the "Codes" tab, input a code in the input window, and then click "OK". You can add an unlimited number of codes, but you should think carefully about what codes are useful in

the research so as to avoid an overwhelming number of codes. You can rename and delete a code from the code list. The operation is similar to that of the files.

When you have imported a list of files and generated a code list, then you can go on to the process of coding. First, you should open a file. Then you read and think, if you think some words are relevant to a code, you can apply that code to the text segment: select a code, select that text segment, and click the "Mark" button in the "Codes" tab. You will see that the text segment is highlighted with a color (by default, it is blue) and a **coding label** has been inserted before the coded segment. We call this process **"coding"**. You can undo the coding as well: (1) for 0.2-1 and previouse versions, select the same text segment and click the "Unmark" button in the "Codes" tab. (2) for 0.2-2, you can click the coding label, and then click the "Unmark" button. Now, the highlighting is removed, which means that you have successfully removed that coding.

If you want to see what information has been attached to a code, you can double click a code to retrieve all the related coding for that code. This process is called **"coding retrieval"**. Of course, you can also select a code, and then click the "Coding" button to perform the same operation. The latter method has the advantage that you can go back to the original file content by clicking the back button within the retrieved coding (though this method takes a bit more time to retrieve the coding). This method can give you a closeness with the original content and avoid, to some extent, segmentation due to coding. To get a descriptive statistics of the coding, you can use the R function summaryCodings() (previously SummaryCodings()). This function will return the "Number of codings for each code", "Average number of words assciated with each code", and "Number of files associated with each code". You can get more by using the getCodingTable() and getFileIds() functions. For more information, please refer to the help page by entering ?getCodingTable and ?getFileIds.

After finishing your coding, you can retrieve the coded text segments with the selected code. Unconditional retrieval, retrieval of codings associated with specific case or file categories are all possible by specifying the type of retrieval (see section **8. setting** for more information).

4. Writing memo

Writing down your thoughts during the reading and coding is an integral part of QDA. This is called "**Memo writing**", and what you write is a "**Memo**". RQDA offers different kinds of memos. For ideas that apply to the whole project, you should write a project memo ("Project Memo" button in the "Project" tab). You can write memo about a specific file by clicking the "Memo" button in the "Files" tab, or by cliking "File Memo" via the pop-up menu in the "Files" tab. In the "Codes" tab the "Memo" button is used to attach memo to a selected code. You can select a code then click "Memo" button to view/add a memo.

5. File categories

File categories help you organize your files. For example, when I analyze newspaper reports, I would like to classify the files accoding to the sources (what newspaper does a file come from), so I add the name of newspapers as a category. You can click the "Add" button in the "File Categories" tab to do so. Then you want to add files to a specific category, you select that category first, then click the "Add To" button, a file list pops up, select files belonging to this category and click "OK". If you want to remove files from a category, select a category first, then files of this category will show in the "Files.of.This.Category", select the files you want to drop from this category, then click the "Drop From" button.

The other way to add files to a category: in the "Files" tab, select the files you want to add to a category, then click "Add to file category ..." in the pop-up menu of the "Files" tab, then a file category list pops up, select one category and click "OK".

Of course, you can rename the file category or delete it. The deletion of the file category will not affect the original files. If you want to delete any of the original files, you should go to the "Files" tab to delete them.

6. Code categories

The code category is a crucial facilitator in **theory development**. The two most important steps in this process are 1) move between concepts of different levels of abstraction; and 2) organize those concepts according to logical relationship between them. Code categories help you to do the second type of work.

First, you add a higher (relative to the codes) level concept by clicking the "Add" button. You can add more code categories by using the same step. Then you add codes to the higher level concept. To do this, you first select that higher level concept, then click the "Add To" button. A list of codes pops up, you can select codes you want to add, and then click "OK". If you want to delete codes from a category, select a category first, then codes of this category will show up in the "Codes.of.This.Categroy", select codes you want to drop from this category, then click the "Drop From" button. The operation is similar to the file categories.

You may want to organize those higher level concepts--**"structure of concepts"**. RQDA doesn't provide a tree-like structure to organize them. I avoid this choice because sometimes tree-like structures create confusion (e.g. loops between concepts are possible, which is the main sources of confusion). Besides, one code can be classified into multiple categories, which is not easily handled by a tree-like structure. Meanwhile, I would like to encourage users to think carefully about the structure of the higher level concepts. After all, theory should inform knowledge in a parsimonious way. It makes no sense to make theory as complex as the real-world itself. Still, it is possible to manage higher level concepts by using proper naming rules. For example, I have "trust" and "norms of reciprocity", and I want to group them by "cognitive social capital". Then I can rename them to "10-trust","10-norms of reciprocity", and "1-cognitive social capital". By sorting (see tip no. 3), you can see the relationship between these concepts.

RQDA provides a basic tool to plot the relationship between codes.

7. Cases

The concept of case in RQDA comes from Qualitative Comparative Analysis (QCA).

The two differences between file category and cases are: a) you can assign attributes to a case but not to a file category. By exporting the attributes, we can perform QCA. b) you can link part of a file to a case but only an entire file to a file category.

How to assign a file to a case: a) select a file in "Files" tab, right-click, in the pop-up menu, you can see "add to a case....". This method will add an entire file to a case. b) Open a file, select part of the file, select a case name, then click button "Link" in "Cases" tab, you can thus link the selected part of file to the selected case.

8. Attributes

"Attribute" is basically the same concept as a variable in quantitative studies. When the "unit of analysis" is clearly defined, then each unit of analysis is treated as a case. You can assign attributes to a file or a to a case. The choice depends on your research design and questions. Usually, attributes of a file are closely related to the data collection process, while the attributes of a case are more analytical since what is a case depends on your analytical process.

For example, the research might be on collective action. Collective action is the unit of analysis, and I treat each as a case. By assigning attributes of collective action I can analyse the conditions of collective action.

To gain information about collective actions, I interview participants. Each interview record is a file, I can assign attributes (e.g. SES of the participants) to each file. The attributes of the files play two roles. First, it can be used to show a subset of files (it can be done by file category, but it is more efficient if the subset condition is complicated). Second, you can do statistics on these SES attributes, so you can report the profile of interviewees.

There is a "Delete" button on the "Attributes" tab, you can select one attribute and delete it. This operation will delete all case and file attributes. So please be cautious.

9.1 Use SQL to perform complex query

There are several very powerful RQDA commands which do not appear in the GUI. To see them enter in the R window: help(package="RQDA")

All kinds of boolean searches can be done on coded data. Unfortunately this has not yet been implemented in the GUI. But boolean searches can be done very easily by typing a command in the R window.

Codes are identified by cid numbers like 1, 2, 3, etc.

To see the cid numbers of different codes give the command: getCodingTable()
To see the selected text for code 1 give the command: RQDAQuery("select seltext from coding where cid=1")
To see codings of code 1 OR code 2, you can search by: RQDAQuery("select seltext from coding where cid=1 or cid=2")

For more on RQDAQuery use: ?RQDAQuery
For more commands use: help(package="RQDA")

Contributed by Amman Madan (amman@iitk.ac.in)

9.2 Boolean search

To perform a complex **Boolean search** programming is needed. Currently, some are supported by RQDA.

To retrieve coding by cid 1 and cid 2, you can run the following command:
coding <- getCodingTable()
c1 <- subset(coding,cid==1)
c2 <- subset(coding,cid==2)
and(c1,c2)

In RQDA 0.2-0, boolean operations are simplified by the functions of %and%, %or% and %not%. To retreive the codings coded by cid 1 and cid 2, you can run: getCodingsByOne(1) %and% getCodingsByOne(2).

To retrieve codings coded by cid 1 but not cid 2, you can run: getCodingsByOne(1) %not% getCodingsByOne(2)

please note that the print method puts the resulting codings into a gtext widget, so you can read it with ease.

You can also use the following commands to perform the same enquiry:
and(getCodingsByOne(1), getCodingsByOne(2))
or(getCodingsByOne(1), getCodingsByOne(2))
not(getCodingsByOne(1), getCodingsByOne(2))

You can do more complex operations as well. For example, to retrieve codings coded by cid 1 and cid 2, plus codings coded by cid 3 but not cid 4. You can run:
summary((getCodingsByOne(1) %and% getCodingsByOne(1)) %or% getCodingsByOne(3) %not% getCodingsByOne(4))

Please note that %and%, %or% and %not% apply to file ids, file names, case ids, case names, all of which are objects with class "RQDA.vector".

9.3. Text mining of codings

By RQDAtm package, codings are easily converted into a Corpus, then you can use the tm package to do text mining. You need to open a RQDA project first. Examples can be found [here](#) [PDF].

10. Settings

"Name of coder" is the name of the analyst. This is crucial if there is more than one analysts, and also crucial for the calculation of the inter-coder reliability coefficient. Thus, you should always set this parameter.

"File Encoding" is the encoding of the file you want to import. If it is an ASCII file, the default "unknown" is fine. If it is another encoding, you should set it accordingly. For example, you should set it to UTF-8 if the file is in UTF-8. Note that if you save a plain text with encoding UTF-8 using MS Notepad, the program will add BOM (Byte order Mark, it is a 3 bite mark) at the beginning of the file, for this case, you should set BOM to TRUE.

It is simple to change a setting. Change the value of each setting and then click the "OK" button. If you want to reset to the default setting, you can just click the "Default" button.

When "Type of Retrieval" is set to "case", and a case is selected, then only codings of the selected case are retrieved. However, if no cases are selected, all codings of the selected code are retrieved. Similarly, when set to "filecategory", and a file category is selected, then only codings of that category will be retrieved.

11. Implementation History

When I started my PhD research project, I planned to conduct some content analysis of newspaper reports. I need a QDA software. I hoped it would have been available to me, have basic functions of QDA software, run both under Windows and Linux/FreeBSD, and last but not least, I should have full control of it. I could not find one, so I decided to write one myself. I had this idea in 2007, but not until late 2008 did I implement most parts of this package.

12. Warranty

As a open source and free project, there is no warranty that it is fit for any particular purpose, or that it will work at all. You use RQDA solely at your own risk, and no liability can be assumed for any damage or loss incurred by your use of the software. You can learn more from the [license](#).

Tips

1. Under Linux/FreeBSD, you should set the locale of R to UTF-8 to get better support of non-English documents. In my FreeBSD box, I set the locale of R by this R command: Sys.setlocale(locale = "zh_CN.UTF-8").
2. When RDQA is launched, it is better to minimize the main window of R (e.g. Rgui, Rterm, terminal from which R is launched).
3. Besides the code categories and file categories, you can help organize files, codes list, and cases by using specific naming rules. For example, if you name a group of codes with 1 as prefix and another group of codes with 2 as prefix, then you can click the triangle symbol in the right-upper corner of the Codes List table and the codes are sorted according to their prefix. Similar trick applies to categories.
4. If a file is large, and you want to quit RQDA without finish coding, you can write a file memo, say "unfinished". Afterwards, you can use "Search File..." of the file pop-up menu to locate the file by "memo like '%unfinished%'".

FAQ

Q: How can I get help?
A: You can join the **[rqda-help](#)** list and post questions via email.
Q: Can I perform statistical analysis of the codes?
A: Depends on the method of coding. For factual coding, you can do statistical analysis. If the documents are representative, you can do referential analysis; otherwise, you can only perform descriptive analysis.
Q: Why should I use computer-aided QDA? Is it more efficient?
A: CAQDA is time-consuming, but it makes the data analysis process more systematic, more tractable, and more transparent. It "forces" you to think carefully about your data and the connection with theory (once you know the idea behind the CAQDA). Depending on the methodological paradigms, you may conduct Qualitative Comparative Analysis or examine some hypotheses. However, not in the sense of statistical test of hypotheses (see Kelle 1995 for details).
Q: My Vista is English, but the *.rqda includes Chinese, and it can not display correctly.
A: One possible reason is a font issue. You can un-install GTK+, launch R, set the locale to CHS by the command Sys.setlocale(,"CHS"), and load RGtk2 using the command library(RGtk2). Then R will ask if you want to install GTK+, and install it. Then you can launch RQDA. It should work.
Q: I press the save button in the memo/journal widget, but it warns that I have not saved it yet when I try to close the widget. Why?
A: This may be due to an encoding issue. Set the appropriate locale using the Sys.setlocale function. The technical aspect is that RQDA needs to convert ' into " using gsub in order to save it in the SQLite database, without proper locales, the conversion may not give you what you want.
Q: Will RQDA support coding of audio files in the near future?
A: Maybe not. I prefer to use specialized software such as [TranscriberAG](#). If necessary, an interface between RQDA and *.trs might be provided in the future.