

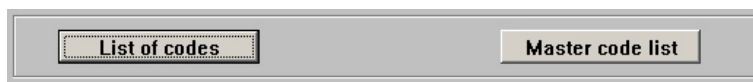
## 10.1 How to retrieve coded text segments and particular codings

There are many good reasons why a researcher now and then may want to get an overview on all those text segments to which the same code was attached. One of these reasons, which is prominent in descriptive/interpretative studies (see Tesch, 1990), is to check commonalities across all data texts. Another reason is to control the consistency of coding. All codes should be supplied always according to the same principles. Still other reasons are to look for locations in the texts where a code was applied correctly vs. erroneously, where similar meanings can be found expressed in different formulations, etc.

AQUAD extracts coded text segments from all texts following the sequence in which they are listed in the file catalog. Because one text after the other is checked sequentially, we call this search for matching text segments a linear or one-dimensional analysis. It is to be distinguished from two-dimensional analyses, called table analyses or matrix analyses (see chapter 11) in AQUAD, and from analyses of complex linkages, also called exploration of linkages (see chapter 11, section 3, and chapter 12). The results can be brought to the screen, to a printer or they can be stored on diskettes.

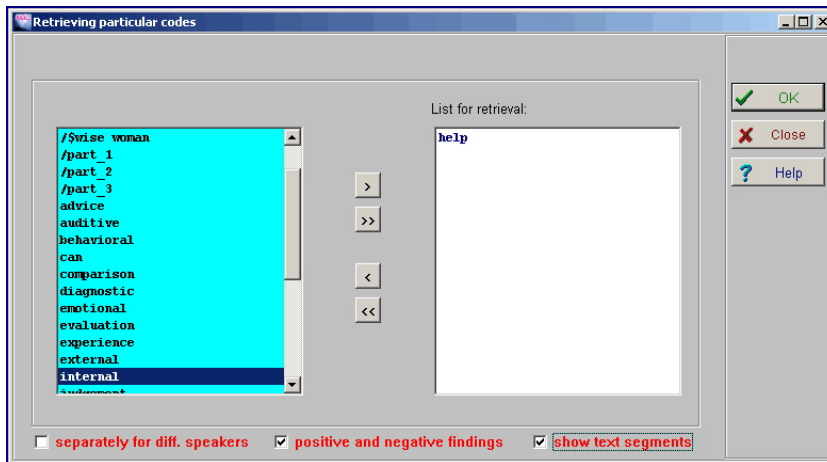
You start a linear search for coded text segments by choosing the option "*Particular codes*" within the module "*Retrieval*." A window opens from which you can decide

- to load one of your already existing code catalogs (i.e., a list of codes) *or*
- to construct a temporary list of codes by selecting the content from the master code list.



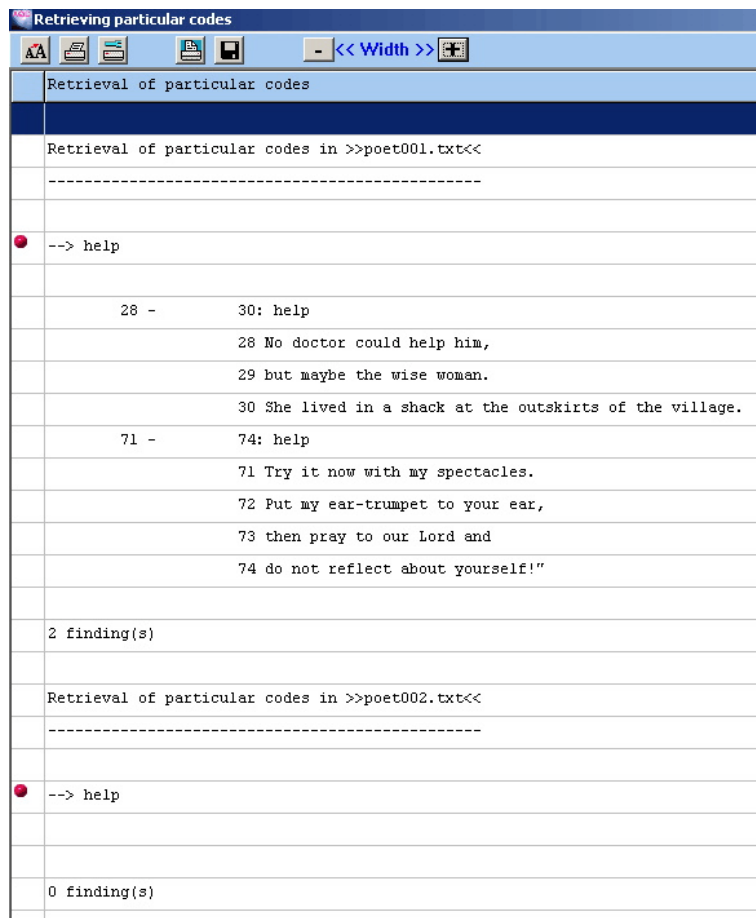
If you select critical codes from the master code list, the codes have to be selected anew whenever you activate this function. In the following example (see screen shot on top of the next page) we select the code "help" from the master code list. Please, pay attention to three check-boxes labeled in red at the bottom of the selection window and mark those options you wish to apply:

- "separately for diff. speakers:" The code/s will be searched for and listed within each datafile separately for different speakers – of course, only if you marked their data segments by speaker codes.
- "positive and negative findings" is marked by default – otherwise missing codes would not appear with zero frequencies in statistical analyses.
- "show text segments" attaches the coded text segment directly to each retrieved code when the results are presented.



If you are sure to use a list of codes for several retrieval runs, you should create a code catalog. To do so you choose – also within the option "Retrieval" – from "Code catalog" the option "Create a code catalog." Then you define a name for saving the catalog you are going to construct. Just enter a name of up to eight characters; the extension ".cco" (catalog of codes) is added automatically. Then a window is opened, where you can create the catalog by selecting critical codes from the master code list. Meanwhile, you should be already familiar with creating catalogs from writing your first file catalog (see chapter 5).

As an example we start a search for all text segments coded as "help" in our sample texts poet.001 - poet.003. The screen shot on the right shows only part of the results:



We will not discuss these findings here, but pay attention to its form: You see that AQUAD shows in which texts (poet.001 through poet.003) it tried to find which the code "help," and that matching text segments are shown together with their line numbers. In case you should want to include these findings later in your research report, you should not forget to click on the diskette icon in the tool bar on top of the window, where the results of the retrieval run are shown (see previous page)..

AQUAD supports permanent comparison of your analytic decisions not only by text retrieval, but by various options for retrieval of codings. We have just seen the first of these options: If you run a search for "*Particular codes*," the program informs you both about the location of text segments to which a particular code was attached and about these segments.

## 10.2 How to retrieve coding structures

Let us assume a researcher has some doubts about several text segments and is not sure which category to apply for coding. Therefore, he or she attached all those codes that appeared to be meaningful to each of these text segments. Now imagine further that in another text two more categories could be applied to a critical text segment. Whenever in doubt, this researcher makes use of AQUAD's possibilities of multiple coding. At some point the researcher gains the insight necessary to clarify the ambiguous codings. It would be very useful now to retrieve all critical text segments quickly, that is without searching for single codes and pick those segments from the numerous findings that were attributed to other categories, too. A problem like that could be easily solved by running a search for coding structures. The researcher would choose "Retrieval" from the main menu, select the option "Coding structures" and activate "Multiple codes" from the next sub-menu.

This sub-menu, appearing after the selection of "Coding structures" offers a choice between several abstract structures of coding. After determining *additionally*, which codes you are trying to retrieve in a particular pattern, you can make AQUAD search for

- nested codes,
- overlapping codes,
- multiple codes,
- sequence of codes, and
- repetition of sequences

in your code files. Here is in more detail what AQUAD will do:

### 10.2.1 Nested codes

With this option the search is aimed for data segments which contain another data segment within the boundaries of its first and last lines. Or formulated differently: all those codings are found, which include within their text area another coding. This option therefore is useful, if you are looking for hierarchically structured sequences of codes. You will not only find codes and sub-codes, but also codings with identical line numbers, as long as their text areas do not overlap (see below).

Retrieval of nested codes: Subordinate codes			
Retrieval of nested codes: Subordinate codes in >>baby_1.avi<<			
-----			
--> right hand			
226 -	747: right hand	/	257 - 272: turn
			287 - 309: turn
			317 - 347: turn
			407 - 604: oral exploration
			619 - 679: turn
1381 -	1433: right hand	/	1396 - 1433: oral exploration
1637 -	1833: right hand	/	1698 - 1833: shaking
2120 -	2195: right hand	/	2120 - 2195: shaking
4 finding(s)			

*Example:* In our sample text "baby" some segments were coded as related to "right hand" movements. Later the coders began to attend to more specific processes like turn, oral exploration, etc. Now we want to merge both perspectives. A search could show, for instance, that particular video scenes were coded as "right hand", while within this segment the code "turn" was attached to a smaller text segment.

AQUAD will report this finding as shown (for the first file "baby\_1" only) in the screen shot above. We see, for instance: Within the super-ordinate segment from frame 226-747, coded "right hand", several turns occurred, for instance between the frames 257-272.

AQUAD is programed to analyze nested codes in two directions:

- You determine a code (or several codes in a code catalog) as superordinated code and AQUAD will retrieve all sub-ordinated codes within the same data segment (as shown in the example on the previous page).
- You determine a sub-ordinated code (or several codes in a code catalog) and AQUAD will retrieve all codes attached to data segments that cover the data segments of this sub-ordinate code.

### 10.2.2 Overlapping codes

The result of this search strategy shows all codes (as always together with their location within a text), which were attached to overlapping text segments. That is, we are informed about both text segments which overlap, not only about the overlapping area.

*Example:* We take our sample texts "Poet" and ask for text segments overlapping all text segments which were coded as "evaluation." We see in the screen shot below only part of the findings, and only from the data file poet003.

Retrieval of overlapping codes			
Retrieval of overlapping codes in >>poet003.txt<<			
-----			
--> evaluation			
8 -	8: evaluation	/	5 - 9: experience
			5 - 9: poet
			8 - 8: negative
			8 - 8: visual
			5 - 9: /\$poet
	5 "What a crowd!" said the young man,		
	6 "one story next to the other.		
	7 It hums and drones!		
	8 That's rather too colorful for me!		
	9 I'm falling backward!"		

Please, keep in mind: "Overlapping" is defined merely in terms of physical overlap of text areas, without any semantic connotations.

The result shows not only codifications, but also the coded data segments, because the check box "show text segments" was marked in the selection window (see section 10.1). But you should be careful with this option: If there are some very comprehensive codings overlapping with many other units of meaning, we would get very long print-outs! Per definition, speaker codes overarch everything within an actual data segment of this speaker! Even more dangerous are profile codes or singular codes, which are valid by definition for the whole data file. Attach them to only to one single line, if possible a blank one at the end – they will be understood nevertheless for the whole file.

### 10.2.3 Multiple codes

This option was already described at the beginning of this chapter. It is used to retrieve all data segments to which more than one code was attached.

*Example:* We assume now that the researcher was not sure whether to code some data segments in the project "poet" generally as dealing with "internal" processes or more specifically as thinking, reflecting, etc. In this situation the researcher decided to apply both categories and to check their validity later. Searching for "multiple codes" including "internal" will produce the following result (here shown only for the data text "poet001" – "show text segments" was marked again):

Retrieval of multiple codes	
Retrieval of multiple codes in >>poet001.txt<<	
-----	
• -->	internal
26 -	27: internal / 26 - 27: think
	26 He tried hard,
	27 until he almost became sick, the poor guy!
60 -	61: internal / 60 - 61: visual
	60 But you do not have the right look on
	61 these things.
62 -	62: internal / 62 - 62: auditive
	62 You do not have a quick ear and
3 finding(s)	

### 10.2.3 Sequence of codes

A very useful first step to approach the reconstruction of linked meanings in a data file is to look for statements, scenes, etc. frequently found in the neighborhood of other units of meaning. We specify a critical code as focus of a search for "sequences of codes." AQUAD will report all combinations of codings used for all data segments which are within a maximal distance – which we determine additionally – of our focus code. "Nested codes" and "overlapping codes" will be reported, too.

*Example:* We want to find out quickly, which other codes were used in the project "poet" in the neighborhood of text segments, to which the code "internal" was attached. As maximal distance we determine 3 lines before or after the critical text segments. That is, AQUAD will report all codes attached to text segments that end within maximally 3 lines before a text segment coded as "internal" begins, and all codes attached to text segments that start within maximally 3 lines after the last line of this critical segment (here you see only part of a very large result file):

Retrieval of code sequences	
Retrieval of code sequences in >>poet001.txt<<	
-----	
Maximal distance 3 units of distance	
• -->	internal
4 -	7: internal <- 1 - 2: \$do not count
	~~ 4 - 9: poetry
	~~ 4 - 47: poet
	-> 8 - 8: must
	-> 8 - 9: rule
	-> 8 - 9: think
	~~ 4 - 15: /\$andersen

### 10.2.4 Repetition of sequences

The main function of AQUAD is to support the endeavors of researchers to find and check relations of meanings in their data files. The program does this with algorithms which apply the principle of "backward deduction." That is, AQUAD takes your assumptions about how codes could be associated and checks all data files to find whether the critical codes can be found in matching sequences (and distances).

Each coded text segment can be conceived of as part of a proof for the corresponding category in your data. While reading your texts, structuring them, comparing them and having now and then a look into your memos you will soon start to develop assumptions about relations between some of the categories used in your analysis. Emerging relations or associations may give rise to the formulation of hypotheses about latent contents of your texts.

The retrieval of *repeated sequences* is a heuristic approach supporting the detection of associations. You enter a code (or several codes) together with the units of measurement (number of lines, seconds, frames) determining a particular area data around its data segments. Whenever a data segment coded with a particular other code name appears *more than once* within this area, AQUAD will report it.

*Example:* We want to find out quickly, whether there are redundant structures including data segments in the project "baby," to which the code "left hand" was attached. As maximal distance we determine  $3 \times 25 = 75$  frames before or after the critical data segments. That is, AQUAD will report all codes attached *more often than once* to data segments that end maximally within 75 frames before a video scene coded as "left hand" begins, and all codes attached to data segments that start within maximally 75 frames after the last frame of this critical segment (here you see only part of a large result file):

Retrieval of repeated sequences					
Retrieval of repeated sequences in >>baby_1.avi<<					
-----					
Maximal distance 75 units of distance					
● --> left hand					
1313 -	1366: left hand	/	1245 -	1283: both hands	
2195 -	2233: left hand	/	2233 -	2241: both hands	
2248 -	2286: left hand	/	2233 -	2241: both hands	
2248 -	2286: left hand	/	2309 -	2324: both hands	
2354 -	2422: left hand	/	2309 -	2324: both hands	
2724 -	3063: left hand	/	3063 -	3139: both hands	
1313 -	1366: left hand	/	1230 -	1275: grasping	
1313 -	1366: left hand	/	1433 -	1456: grasping	
1426 -	1479: left hand	/	1433 -	1456: grasping	
1494 -	1599: left hand	/	1433 -	1456: grasping	
1494 -	1599: left hand	/	1599 -	1630: grasping	

Remember: These findings may give us a hunch about linkages in our data files, which may be meaningful. However, we have to make sense of them – or to reject them as meaningless repetitions of codes.

### 10.3 Digression: Distance between data segments

Hypotheses about linkages of codes (or more precisely: linkages of coded data segments) make sense in most cases only if we include a formulation like "... under the condition of a specific maximal distance between them." Take, for instance, the example of the project "interview" (on your hard disk) and suppose we got a hunch that some speakers (teachers) begin to reflect about themselves in these interviews after they talked about a typical dilemma in their classrooms. In terms of a linkage hypothesis: We observe that a speaker talks in a particular data segment about a "dilemma" – and now we expect that he/she talks in the following data segment about himself/herself (coded as "SIM self" in the example).

Since the program is unable to analyze semantically, whether there is a relation of meaning or not, we have to rely as approximative solution to limit the distance between critical segments in the stream of words. Most probably there is no direct relation between two instances of our critical codes, if a teacher talks about conflicting tendencies of reacting to students, for instance, allowing space for spontaneity vs. insisting strictly on rules, and ten minutes (or four pages in the transcription) later he/she talks about being somewhat frustrated. If there is a direct relation, most speakers will refer explicitly again to an idea (in our example: a cause) they expressed already earlier. We would have to code this – maybe very short – reference once more as "dilemma." The default distance is set to three "units", that is in case of text analysis three lines as maximal distance. Data segments "SIM self" beginning only four lines after the end of a segment coded as "dilemma" would not be registered as positive findings in our example. What follows is that we have to experiment with the distance settings and adapt it to the characteristics of our data.

The following screen shot uses an example from "two-step coding" the files of the project "poet" (was copied to your harddisk during installation): The data segment

```
He knew, all you need to do is
to think up something,
...
About everything was already written
poetry.
```

was coded as "B." After a blank line follows a data segment "Those happy people ..." coded as "X." The default setting  $d=3$  (maximal distance = 3 lines) would report a positive result for a hypothetical linkage of B-X, however a linkage of F-X would not be reported in this case.

The screenshot shows two text segments with handwritten annotations. The first segment is coded 'B' and the second 'X'. A bracket indicates a distance of  $d=3$  between the end of 'B' and the start of 'X'.

Segment B (top):

```
He knew, all you need to do is
to think up something,
but he was unable to imagine anything.
He came into the world too late,
everything was already done
before he was born.
About everything was already written
poetry.
```

Segment X (bottom):

```
"Those happy people, who were
born thousand years ago," he said,
"it was easy for them to become immortal!
Lucky also, who was born
hundred years ago,
then there was still something to versify;
now the world is emptied by poets,
what should I compose into it today?"
```

Handwritten annotations:

- A bracket labeled 'B' spans the first segment.
- A bracket labeled 'X' spans the second segment.
- A bracket labeled  $d=3$  indicates the distance between the end of segment B and the start of segment X.
- A bracket labeled  $d=3$  indicates the distance between the end of segment X and the start of the next segment (not fully visible).

---

The meaning of "distance" between text segments should be clear: "d" stands for the maximal number of lines between the end of one segment and the beginning of another segment that are hypothetically linked together. But what is the meaning of "distance" in audio recordings or scenes in a video file?

**Audio files:**

The mediaplayer counts the audio stream in 1/10 of a second. The distance value in retrieval runs or linkage hypotheses is multiplied by 10, that is, distances are defined in seconds. The default setting  $d=3$  causes then that audio segments beginning maximally 3 seconds after a previous segment are accepted as "related" to this segment.

**Video files:**

The mediaplayer counts the video stream in frames, that is, single pictures. The distance setting is multiplied by 25, that is, according to the PAL norm of 25 frames per second, the distance is checked in seconds again. The default setting  $d=3$  causes then that video segments beginning maximally 3 seconds (or 75 frames) after a previous segment are accepted as "related" to this segment.

#### 10.4 How to learn about unused codes

This option in the menu "*Retrieval*" starts a one-dimensional search. It is often quite helpful in a large project to learn which of the many codes were *not* used in connection with which texts. AQUAD takes the momentary content of the master code file (see chapter 6, section 8) or any selection of codes as criterion for this type of search.

---

## 10.5 How to retrieve codes, count them and enter their frequencies into tables

Information about the frequencies of selected codes of a project gives access to various quantitative analyses, which may be really meaningful – depending on the research question (see chap. 14). You may create frequency tables for codes in AQUAD and import them later into software for quantitative analysis like spreadsheets or statistical programs. This is true also for sequential codes, which are applied by AQUAD automatically during linkage analyses (see below, chapter 11, sections 11.3 and 11.4).

In any case you start by having AQUAD "*Count codes*," one of the options in "*Retrieval*" in the main menu. AQUAD summarizes the results in a simple frequency list (listing of counted codes and their frequencies separately for each data file in your project). You save this list under any name in the subdirectory `..\res` by clicking on the symbol of a diskette in the tool bar on top of the result window.

Statistic programs are at a loss with listings of this type, therefore we have to convert them into tables structured by codes (=variables) filling the columns and cases (files) defining the rows. The cells of these tables then contain frequencies for each (selected) code in each of the cases of our project. There is a specific conversion routine within the options of "*Implicants*" (see chap. 13), and a general tool, of course, in the group "*Tools*" for converting lists into tables.

There are two alternatives for saving the resulting tables, which you can apply both for any table: (1) Code frequencies are stored in AQUAD's proprietary table format ".adt" (AQUAD data table) and can be opened directly for table analyses (see chapter 11, section 11.2) – if the number of codes does not exceed the limits for this sort of analysis; (2) the code frequencies are separated by commas and saved line by line, resulting in a "csv" table (comma separated values), which can be opened directly in spreadsheet programs (for instance, QuattroPro) or in statistical software packages (for instance, SPSS). Details of how to apply this tool can be found in chapter 15.

