ATLAS.ti
QUALITATIVE DATA ANALYSIS

Inter-Coder Agreement Analysis

ATLAS.ti 8 Mac

ATLAS.ti 8 Mac - Inter-Coder Agreement Analysis

Document version: 751.20200507.

Author: Dr. Susanne Friese (QuaRC)
Production: hypertexxt.com/Dr. Thomas G. Ringmayr

Contents

Inter-coder Agreement (ICA)

Why It Matters

The purpose of collecting and analyzing data is that researchers find answers to the research questions that motivated the study in the first place. Thus, the data are the trusted ground for any reasoning and discussion of the results. Therefore, the researchers should be confident that their data has been generated taking precaution against distortions and biases, intentional or accidental, and that the mean the same thing to anyone who uses them. Reliability grounds this confidence empirically (Krippendorff, 2004).

Richards (2009) wrote: "*But being reliable (to use the adjective) beats being unreliable. If a category is used in different ways, you will be unable to rely on it to bring you all the relevant data. Hence, you may wish to ensure that you yourself are reliably interpreting a code the same way across time, or that you can rely on your colleagues to use it in the same way*" (p. 108).

There are two ways to rationalize reliability, one routed in measurement theory, which is less relevant for the type of data that ATLAS.ti users have. The second one is an interpretivist conception of reliability. When collecting any type of interview data or observations, the phenomena of interest usually disappears right after it has been recorded or observed. Therefore, the analyst's ability to examine the phenomena relies heavily on a consensual reading and use of the data that represent the phenomena of interest. Researchers need to presume that their data can be trusted to mean the same to all of their users.

This means "*that the reading of textual data as well as of the research results is replicable elsewhere, that researchers demonstrably agree on what they are talking about. Here, then, reliability is the degree in which members of a designated community agree on the readings, interpretations, responses to, or uses of given texts or data. **[...]** Researchers need to demonstrate the trustworthiness of their data by measuring their reliability*" (Krippendorff, 2004, p. 212).

Testing the reliability of the data is a first step. Only after establishing that the reliability is sufficiently high, it makes sense to proceed with the analysis of the data. If there is considerable doubt what the data mean, it will be difficult to justify the further analysis and also the results of this analysis.

ATLAS.ti's inter-coder agreement tool lets you assess the agreement of how multiple coders code a given body of data. In developing the tool we worked closely together with Prof. Klaus Krippendorff one of the leading experts in this field, author of the book Content Analysis: An Introduction of Its Methodology, and the originator of the Krippendorff's alpha coefficient for measuring inter-coder agreement.

The need for such a tool as an integrated element in ATLAS.ti has long been evident and has been frequently requested by users. By its nature, however, it could not and cannot be a magic "just click a button and hope for the best" solution kind of tool. If you randomly click on any of the choices that ATLAS.ti offers to calculate an inter-coder agreement coefficient, ATLAS.ti will calculate *something*. Whether the number you receive will be meaningful and useful depends on how you have set up your project and your coding.

This means if you want to test for inter-coder agreement, it requires at least a minimal willingness to delve into some of the basic theoretical foundations of what inter-coder agreement is, what it does and can do, and also what it cannot do. In this manual, we provide some of the basics, but this cannot be a replacement for reading the literature and coming to understand the underlying assumptions and requirements for running an inter-coder agreement analysis.

Please keep in mind that the inter-coder agreement tool crosses the qualitative-quantitative divide. Establishing inter-coder agreement has its origin in quantitative content analysis (see for instance Krippendorff, 2019; Schreier, 2012). If you want to apply it and want to adhere to scientific standards, you must follow some rules that are much stricter than those for qualitative coding.

If you want to develop a code system as a team, yes, you can start coding independently and then see what you get. But this approach can only be an initial brainstorming at best. It cannot be used for testing inter-coder agreement.

At What Phase in Your Project Should ICA Analysis be Performed?

A good time to have your coding checked by other coders is when you have built a stable code system and all codes are defined. This means, this is somewhere in the middle of the coding process. Once a satisfactory ICA coefficient is achieved, the principal investigator has the assurance that his or her codes can be understood and applied by others and can continue to work with the code system.

Reliability and Validity

Whereas reliability offers the certainty that research findings can be reproduced and that no or only limited external "noise" has contaminated the data or the results, validity assures that the assertions made by the research reflect the reality it claims to represent. Validity concerns truth(s).

Reliability relates to validity in the following ways:

The more unreliable the data, the less likely it is that researchers can draw valid conclusions from the data. In terms of coding data, this means that researchers need to identify valid accounts in the data to a degree better than by chance. If the agreement of two or more coders is not better than the agreement by chance, then reliability is low and you cannot infer that a common understanding of the data exists. Thus: **Unreliability limits the chance of validity.**

On the other hand, **reliability does not necessarily guarantee validity**. Two coders may share the same world view and have the same prejudices may well agree on what they see, but could objectively be wrong. Also, if two researchers may have a unique perspective based on their academic discipline but their reading is not shared by many people outside their own scholarly community, the reliability might be high but the outcome of the research has little chance of being substantiated by evidence of the reality that is inferred. As Krippendorff (2004) states: "Even perfectly dependable mechanical instruments, such as computers, can be wrong – reliably." (p. 213).

A third aspect is the **dilemma between high reliability and validity**. Interesting interpretations might not be reproducible, or interesting data may not occur often enough to establish reliability. Highly reliable data might be boring and oversimplified in order to establish a high reliability in the first place.

Requirements for Coding

Sources for unreliable data are intra-coder inconsistencies and inter-coder disagreements. To detect these, the coding process needs to be replicated. Replicability can be assured when several independently working coders (at least two) agree:

- on the use of the written coding instruction.
- by highlighting the same textual segments to which the coding instructions apply.
- by coding them using the same codes, or by identifying the same semantic domains that describe them and code them using the same codes for each semantic domain.

Semantic Domains

A semantic domain is defined as a set of distinct concepts that share common meanings. You can also think about them as a category with sub codes. Examples of semantic domains are:

EMOTIONS
emotions: joy
emotions: excitement
emotions: surprise
emotions: sadness
emotions: anger
emotions: fear

STRATEGIES
strategies: realize they are natural
strategies: have a Plan B
strategies: adjust expectations
strategies: laugh it off
strategies: get help

ACTOR
actor: partner
actor: mother
actor: father
actor: child
actor: sibling
actor: neighbour
actor: colleague

Each semantic domain embraces **mutually exclusive concepts** indicated by a code. If you code a data segment with 'emotions: surprise', you cannot code it also 'emotions: fear'. If both are mentioned in close proximity, you need to create two quotations and code them separately. You can however apply codes from different semantic domains to one quotation. You find more information on **multi-valued coding** in the section "Rules for Applying Codes".

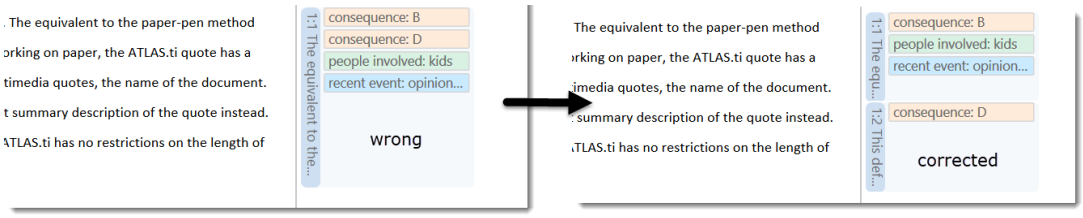

Figure 1: How to correct a violation of mutual exclusiveness

Semantic Domains are Context Dependent

At times it might be obvious from the domain name what the context is. The above sub code 'supporting each other' belongs to the context 'benefits of friendship' and it is not about work colleagues supporting each other. If the context is still unclear and could be interpreted in different ways, you need to make it unambigious in the code definition.

Semantic Domains need to be Conceptually Independent

Conceptual independence means:

- a sub code from one domain is only specific for this domain. It does not occur in any other domain.
- thus, each code only occurs once in the code system

Therefore, as semantic domains are logically or conceptually independent from each other, it is possible to apply codes from different semantic domains to the same or overlapping quotations. For instance, In a section of your data, you may find something on the benefits of friendship and some emotional aspects that are mentioned. As these codes come from different semantic domains (BENEFITS and EMOTIONS), they can both be applied.

Developing a Code System with Semantic Domains

Semantic domains can be developed deductively or inductively. In most studies applying a qualitative data analysis approach, development is likely to be inductive. This means, you develop the codes while you read the data step by step. For example, in an interview study about friendship, you may have coded some data segments 'caring for each other', 'supporting each other' and 'improve health and longevity'. Then you realize that these codes can be summarized on a higher level as 'BENEFITS OF FRIENDSHIP'. Thus, you set up a semantic domain BENEFITS with the sub codes:

- benefits: caring for each other
- benefits: supporting each other
- benefits: improve health and longevity

You continue coding and come across other segments that you code 'learning from each other'. As this also fits the domain BENEFITS, you add it to the domain by naming it:

- benefits: learning from each other

And so on. This way you can build semantic domains step by step inductively while developing the code system.

Once the code system is ready and the code definitions are written in the code comment fields, you can prepare the project for inter-coder agreement testing. At this stage, you can no longer expand a semantic domain. See "Project Management for the Principal Investigator (PI)".

When developing a code system, the aim is to cover the variability in the data so that no aspect that is relevant for the research question is left-out. This is referred to as **exhaustiveness**. On the domain level this means that all main topics are covered. On the sub code level, this means that the codes of a semantic domain cover all aspects of the domain and the data can be sorted in the available codes without forcing them. An easy way out is to include a catch all 'miscellaneous' code for each domain into which coders can add all data that they think does not fit anywhere else. However, keep in mind that such catch all codes will contribute little to answering the research questions.

Rules for Applying Codes

- codes from one domain need to be applied in a mutually exclusive manner.
- codes from multiple semantic domains can be applied to the same or overlapping data segments.
- **Mutual exclusiveness:** You can only apply one of the sub codes of a semantic domain to a quotation or to overlapping quotations. Using the same code colour for all codes of a semantic domain will help you to detect possible errors.

If you find that you have coded a quotation with two codes from the same semantic domain, you can fix it by splitting the quotation. This means, you change the length of the original quotation and create one new quotation, so you can apply the two codes to two distinct quotations. See Figure 1: How to correct a violation of mutual exclusiveness.

If codes within a semantic domain are not applied in a mutually exclusive manner, the cu-alpha coefficient cannot be calculated.

Multi-Valued Coding: This means that you can apply multiple codes of different semantic domains to the same quotation. For instance, a respondent talks about anger in dealing with her mother and mentions that she had to adjust expectations, this can be coded with codes from the three semantic domains EMOTIONS; ACTOR and STRATEGIES.

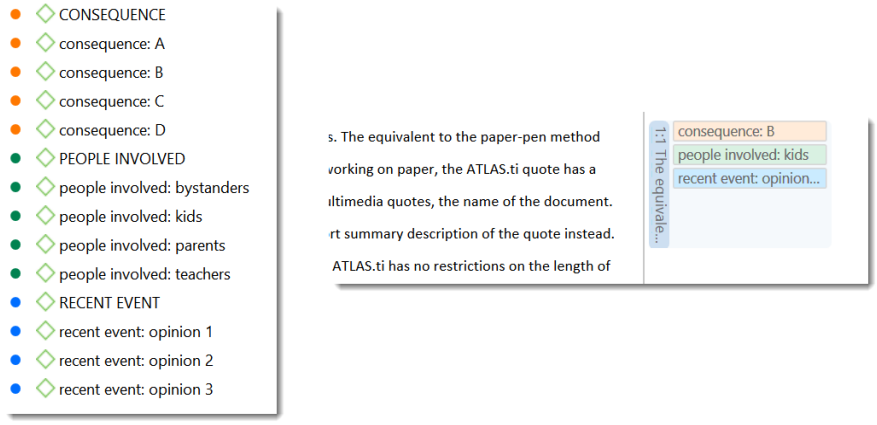

Figure 2: Mulit-valued coding

Coding with codes from multiple semantic domains will allow you to see how the various semantic domains are related for other analyses than inter-coder agreement, the example between opinions on recent events, people involved and type of consequences. For this you can use the code co-occurrence tools. You find information on the Code Co-Occurrence Tools in the full manual.

Instructions for Coders (Summary Form)

An important requirement for inter-coder agreement analysis is the independence of the coders. Thus, the person who has developed the code system cannot be one of the coders whose coding goes into the ICA analysis. In addition to the principal investigator who develops the code system, two or more persons are needed who apply the codes.

The coders will receive a project bundle file from the principle investigator (see "Project Management for the Principal Investigator (PI)"). It is recommended to provide the coders with the following instruction:

- After you imported the project that you received from the PI, rename the project and add your name or initials to the project name.
- Ddouble check your user name. See "User Accounts".
- Apply all codes of a semantic domain in a mutual exclusive manner. See "Rules for Applying Codes".
- If fitting you can apply codes from multiple semantic domains to the same or overlapping quotations.
- Do not make any changes to the codes - do not alter the code definition, do not change the code name or the code color.
- If you do not understand a code definition, or find a code label not fitting, create a memo and name it 'Comments from coder name'. Write all of your comments, issues you find and ideas you have into this memo.
- Do not consult with other coders. It is important that your coding remains unbiased and the data is coded by all coders independently.
- Once you are done coding, create a project bundle file and send it back to the PI. See"Exporting the Project for the PI".

How Agreement / Disagreement Is Measured

The coefficients that are available measure the extent of agreement or disagreement of different coders. Based on this measure one can infer reliability, but the coefficients do not measure reliability directly. Therefore: what is measured is inter-coder agreement and not inter-coder reliability.

Agreement is what we measure; reliability is what we infer from it.

Suppose two coders are given a text and code it following the provided instructions. After they are done, they give you the following record of their coding:

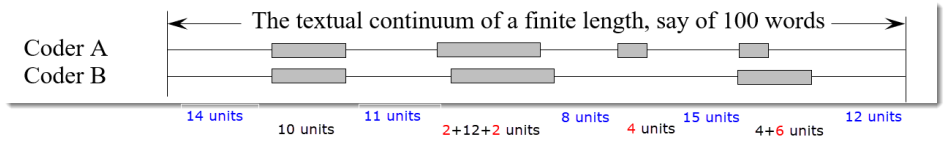

Figure 3: Textual continuum

- There is agreement on the first pair of segments in terms of length and location (agreement: 10 units).
- On the second pair they agree in length but not in location. There is agreement on an intersection of the data (agreement; 12 units, disagreement: 2 + 2 units).
- In the third pair, coder A finds a segment relevant that is not recognized by coder B.(disagreement: 4 units).
- The largest disagreement is observed in the last pair of segments. Coder A takes a narrower view than coder B (agreement: 4 units, disagreement: 6 units).

In terms of your ATLAS.ti coding, you need to think of your document as a textual continuum. Each character of your text is a unit of analysis for ICA, starting at character 1 and ending for instance at character 17500. For audio and video documents, the unit of analysis is a second. Images can currently not be used in an ICA analysis.

The quotation itself does not go into the analysis, but the characters or seconds that have been coded. In other words, if each coder creates quotations, it is not a total disagreement if they have not highlighted the exact same segment. The part of the quotations that overlap go into the analysis as agreement; the other parts as disagreement.

Another option is to work with pre-definded quotations. So that coders only need to apply the codes. For more information see “Project Management for the Principal Investigator (PI)“.

You can use the inter-coder agreement tool for text, audio and video data.

Methods for Testing ICA

ATLAS.ti currently offers three methods to test inter-coder agreement:

- Simple percent agreement
- Holsti Index
- Two of the Krippendorff's family of alpha coefficients.

All methods can be used for two or more coders. For scientific reporting, we recommend the Krippendorff alpha coefficients.

Percent Agreement

Percentage Agreement is the simplest measure of inter-coder agreement. It is calculated as the number of times a set of ratings are the same, divided by the total number of units of observation that are rated, multiplied by 100.

The benefits of percentage agreement are that it is simple to calculate and it can be used with any type of measurement scale. Let's take a look at the following example: There are ten segments of text and two coders only needed to decide whether a code applies or does not apply:

Segments	1	2	3	4	5	6	7	8	9	10
Coder 1	1	1	0	0	0	0	0	0	0	0
Coder 2	0	1	1	0	0	1	0	1	0	0

Percent Agreement (PA) = number of agreements / total number of segments

PA = 6 / 10

PA =0.6 = 60%

Coder 1 and 2 agree 6 out of 10 times, so percent agreement is 60%. One could argue this is quite good. This calculation, however, does not account for chance agreement between ratings. If the two coders were not to read the data and would just randomly code the 10 segments. we would expect them to agree a certain percentage of the time by chance alone. The question is: How much higher is the 60% agreement over the agreement that would occur by chance? Below only the results are presented if chance agreement is taken into account. If you are interested in the calculation, take a look at Krippendorff (2004, p. 224-226). The agreement that is expected by mere chance is (9.6 +1.6)/20 = 56%. The 60% agreement thus is not impressive at all. Statistically speaking, the performance of the two coders is equivalent to having reliably coded only 1 of the 10 segments, and have arbitrarily assigned 0s and 1s to the other 9 segments.

Holsti Index

Holsti's method (1969) is a variation of percentage agreement, as percent agreement cannot be used if the coders have not all coded the same data segments. When coders were allowed to create their own quotations and did not code pre-defined quotations, the Holsti index needs to be user. P'lease note, that also the Holsti index does not take into account chance agreement.

The formula for the Holsti Index is:

PA (Holsti) = 2A/ (N1+N2)

PA (Holsti) represents percentage of agreement between two coders,

A is the number of the two coders' consensus decisions, and N1 and N2 are numbers of decisions the coders have made respectively .

Percentage agreement and the Holsti Index are equal when all coders code the same units of sample.

Cohens Kappa

ATLAS.ti does not offer a calculation for Cohen's Kappe, because of severe limitations. Cohen's kappa is a modification of Scott's pi and according to Krippendorff (2019) and Zwick (1988) a rather unfortunate one because of a conceptual flaw in its calculation. Unlike the more familiar contingency matrices, which tabulate N pairs of values and maintain reference to the two coders, coincidence matrices tabulate then pairable values used in coding, regardless of who contributed them, in effect treating coders as interchangeable. Cohen's kappa, by contrast, defines expected agreement in terms of contingencies, as the agreement that would be expected if coders were statistically independent of each other. Cohen's conception of chance fails to include disagreements between coders' individual predilections for particular categories, punishes coders who agree on their use of categories, and rewards those who do not agree with higher kappa-values. This is the cause of other noted oddities of kappa.The statistical independence of coders is only marginally related to the statistical independence of the units coded and the values assigned to them. Cohen's kappa, by ignoring crucial disagreements, can become deceptively large when the reliability of coding data is to be assessed.

In addition, both Cohen's kappa and Scott's P assume an infinite sample size. Krippendorff's alpha coefficient in comparison is sensitive to different sample sizes and can also be used on small samples.

Krippendorff's Family of Alpha Coefficients

The family of alpha coefficients offers various measurement that allow you to carry out calculations at different levels:

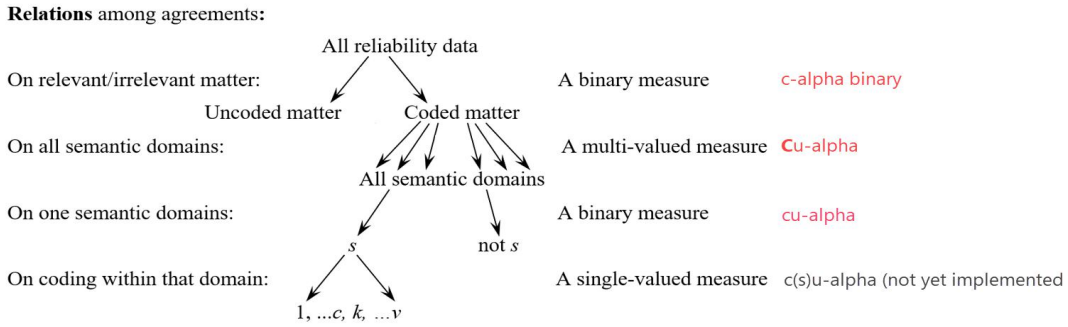

Figure 4: Krippendorff's alpha family – from the general to the specific

C-ALPHA BINARY

At the most general level, you can measure whether different coders identify the same sections in the data to be relevant for the topics of interest represented by codes.

You can, but do not need to use semantic domains at this level. It is also possible to a enter single codes per domain. You get a value for alpha binary for each code or semantic domain in the analysis, and a summary value for all items in the analysis.

All text units are taken into account for this analysis, coded and uncoded matter.

If you work with pre-defined quotations, the binary coefficient will be 1 for a semantic domain if only codes of the same semantic domain have been applied, regardless of which codes within the domain.

The summary alpha binary is always 1 when working with pre-defined quotations, as all coded segments are the same.

CU-ALPHA

Another option is to test whether different coders were able to distinguish between the codes of a semantic domain. For example, if you have a semantic domain called 'type of emotions' with the sub codes:

- emotions::contentment
- emotions::excitement
- emotions::embarrassment
- emotions::relief

The coefficient gives you an indication whether the coders were able to reliably distinguish between for instance 'contentment' and 'excitement', or between 'embarrassment' and 'relief'. The cu-alpha will give you a value for the overall performance of the semantic domain. It will however not tell you which of the sub codes might be problematic. You need to look at the quotations and check where the confusion is.

The cu-alpha coefficient can only be calculated if the codes of a semantic domain have been applied in a mutually exclusive manner. This means only one of the sub codes per domain is applied to a given quotation. See "Rules for Applying Codes".

Cu-alpha

Cu-alpha is the summary coefficient for all cu-alphas. It takes into account that you can apply codes from multiple semantic domains to the same or overlapping quotations. See the information on multivalued coding in the section "Requirements for Coding". Thus Cu-alpha is not just the average of all cu-alphas.

If codes of a semantic domain A have been applied to data segments that are coded with codes of a semantic domain B, this does not affect the cu-alpha coefficient for either domain A or B, but it effects the overall Cu-alpha coefficient. You can interpret the Cu-alpha coefficient as indicating the extent to which coders agree on the presence or absence of the semantic domains in the analysis. Formulated as a question: Could coders reliably identify that data segments belong to a specific semantic domain, or did the various coders applied codes from various other semantic domains?

In the calculation for both the cu- and Cu-alpha coefficient, only coded data segments are included in the analysis.

C(s)U-alpha

This coefficient also belongs to the family of alpha coefficients, but it is not yet implemented. Once implemented, it will allow you to drill down a level deeper and you can check for each semantic domain which code within the domain performs well or not so well. It indicates the agreement on coding within a semantic domain.

For example, if you have a domain 'type of emotions' with the sub codes:

- emotions::contentment
- emotions::excitement
- emotions::embarrassment
- emotions::relief

Coders may not code 'contentment' and 'excitement' consistently. You may have a cu-alpha coefficient for the entire domain of 0,76, which is satisfactory (see "Acceptable Levels of Reliability"). But if you look inside the semantic domain, the coefficients for 'contentment' and 'exciement' might not be so high. Thus, the c(s)u-alpha coefficient allows you to look inside a semantic domain.

As background reading we recommend: Krippendorff, Klaus (2019). Content Analysis: An Introduction to its Methodology. Thousand Oaks, 4th edition. California: SAGE publications.

Calculation of Krippendorff's Alpha

According to Krippendorff (2019), the inter-coder agreement can be calculated as follows:

$$\alpha = 1 - \frac{D_o}{D_e}$$

D_o: observed disagreement; D_e: expected disagreement

whereby De is the expected disagreement by chance.

α = 1 indicates perfect reliability.

α = 0 indicates the absence of reliability. The units and the values assigned to them are statistically unrelated.

$\alpha < 0$ when disagreements are systematic and exceed what can be expected by chance.

When coders pay no attention to the text, which means they just apply the codes in an arbitrary manner, their coding has no relationship to what the text is about, then the observed disagreement is 1. In other words, the observed disagreement (Do) equals the maximum expected disagreement (De), which is 1. If we enter this in the formula, we get:

α= 1 – 1/1 = 0.000.

If on the other hand, agreement is perfect, which means observed disagreement (Do) = 0, then we get:

α= 1 – 0/expected disagreement = 1.000.

To better understand the relationship between actual, observed and expected agreement, let's look a the following contingency table:

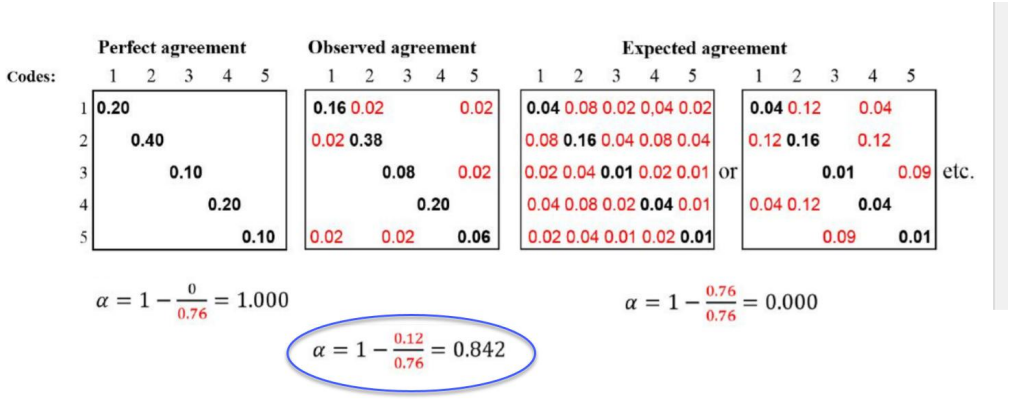

Figure 5: Contingency table for a more complex example

The tables represent a semantic domain with five codes. The matrices with perfect and expected agreement/disagreement serve as a benchmark what could have been if agreement either would have been perfect or if coders had applied the codes randomly. There are two matrices for expected agreement/disagreement indicating that statistically speaking there are many possibilities for a random attribution of the codes.

The black numbers show agreement, the red numbers disagreement.

- In the contingency table for observed agreement, we can see that the coders agree in applying code 4. There are no red numbers, thus no disagreement.
- There is some disagreement about the application of code 2. Agreement is 38% percent. In 2 % of the caes, code 1 was applied instead of code 2.
- There is a bit more confusion regarding the application of code 5. The coders have also applied code 1 and code 3.

If this were a real life example, you would take a look at the code definitions for code 5 and code 2 and ask yourself why coders had difficulties to apply these codes in a reliable manner. The coefficient, however, is acceptable (see "Acceptable Levels of Reliability"). There is no need for re-testing.

For a full mathematical explanation on how the alpha coefficients are calculated see the Appendix written by Prof. Krippendorff and Wikipedia.

Common Mistakes

A common belief is that **consensus is better than individual judgment**. Rather than to work independently, coders are encouraged to discuss what they code and to reach their decision by compromise. However, data generated in this way does not ensure reproducibility, nor does it reveal the extent of it. The results often reflect the social structure of the group, the most prestigious member of the group dominating the outcome. Reproducibility requires at least two independent coders.

Another common procedure is **to allow coders to consult each other** if problems arise, e.g. they do not understand a coding instruction, or they have problems applying some of the codes to the data given to them. This also compromises reliability. Ideally, the coding instructions should be clear and easy to understand. If this is not the case, the coders involved in the discussion create their own interpretation of what the codes mean. This is difficult to communicate to others and therefore jeopardizes reproducibility. In addition, the process loses stability as the data coded in the early phases were coded based on a different understanding of the codes. If coders do not work independently and discuss problems during the process of coding, the reliability coefficient might be higher in the end, but this is partly illusory. If the data were to be given to different coders not having the same insights, reliability is likely to be lower again. As it is common that code systems evolve, and code definitions and coding rules need to be refined, you can ask coders to write down all the issues they see with the current instructions and definitions. Based on their notes, the coding system can be refined and tested again, but with different coders.

Another common mistake is ***to assume that it is best to ask other experts, colleagues with a long history of involvement in the subject of the research, or close friends and associates*** *to serve as coders. Those coders are likely to agree, but not because they carefully follow the coding instructions, but because they know each other and the purpose of the research. This also results in higher measures of reliability, but also does not serve reproducibility.*

Sample Size

The data you use for the ICA analysis needs to be representative of the total amount of data you have collected, thus of those data whose reliability is in question. Furthermore, the number of codings per code needs to be sufficiently high. As a rule of thumb, the codings per code should at least yield five agreements by chance. Krippendorff (2019) uses Bloch and

Kraemer's formula 3.7 (1989:276) to obtain the required sample size. The table below lists the sample size for the three smallest acceptable reliabilities α_{min}: 0.667 / 0.800 / 0.900; for four levels of statistical significance: 0.100 / 0.050 / 0.010 / 0.005, and for semantic domains up to10 codes (probability $p_{c)}$:

Smallest acceptable α	**.667**				**.800**				**.900**			
Significance level	.100	.050	.010	.005	.100	.050	.010	.005	.100	.050	.010	.005
codes per semantic domain												
10 codes or pc = .100	104*	172	344	421	178	293	587	719	361	595	1190	1459
9 codes or pc = .111	95	156	312	383	162	267	534	654	329	542	1083	1328
8 codes or pc = .125	85	141	281	345	146	241	481	590	297	489	980	1198
7 codes or pc = .143	76	125	251	307	130	214	429	526	265	436	872	1069
6 codes or pc = .167	67	110	220	270	114	189	377	462	233	384	768	941
5 codes or pc = .200	58	95	190	233	99	163	326	400	202	332	667	815
4 codes or pc = .250	49	81	161	198	84	139	277	340	172	283	566	694
3 codes or pc = .333	41	67	135	165	71	116	233	285	144	238	477	584
2 codes or pc = .500	36	60	119	146	62	103	206	252	128	211	422	518

*number of codings for the semantic domain after merging

Table 1: Required sample size (adapted from Krippendorff, 2018)

Example: If you have a semantic domain with 4 codes and each of the codes are equally distributed, you need a minimum of 139 codings for this semantic domain if the minimum alpha should be 0.800 at a 0.05 level of statistical significance. For a lower alpha of 0.667, you need a minimum of 81 codings at the same level of statistical significance.

The formula for calculating the sample size is as follows:

$$N_c = 2 * Z_p^2 \left(\frac{(1+\alpha_{min})(3-\alpha_{min})}{4(1-\alpha_{min})P_c(1-P_c)} - \alpha_{min} \right)$$

Figure 6: Calculating sample size if your semantic domains have more than 10 codes and you work with 2 coders

It applies to 2 coders and for semantic domains up to 10 sub codes.

- By 4 codes in a semantic domain, the estimated proportion p_c of all values c in the population is 0.250 (¼), by 5 codes, it is 0,20 (1/5), by 6 codes 0,1667 (1/6) and so on. If the distribution of your codes in a semantic domain is unequal, you need to make a new estimate for the sample size by using a p_c in the formula shown in Figure 2that is correspondingly less than 1/4, 1/5, 1/6 and so on.
- If your semantic domain has more than 10 codes, you can calculate the needed sample size by adjusting the value for p_c as well.

You can see the corresponding z value from a standard normal distribution table. For a p-value of 0,05, z = 1,65.

The equation to estimate the needed sample size applies when working with 2 coders. If you work with more coders, you need to increase the sample size accordingly.

Acceptable Levels of Reliability

The last question to consider is when to accept or reject coded data based on the ICA coefficient. Krippendorff's (2019) recommendations are:

- Strive to reach α = 1,000, even if this is just an ideal.
- Do not accept data with reliability values below α = 0.667.
- In most cases, you can consider a semantic domain to be reliable if α ≥ 0.800.
- Select at least 0.05 as the statistical level of significance to minimize the risk of a Type 1 error, which is to accept data as reliable if this is not the case.

Which cut-off point you choose always depends on the validity requirements that are placed on the research results. If the result of your analysis affects or even jeopardizes someone's life, you should use more stringent criteria. A cut-off point of α

= 0.800 means that 80% of the data is coded to a degree better than chance. If you are satisfied with α= 0.500, this means 50% of your data is coded to a degree better than chance. The

How To Set Up a project for ICA Analysis

When you prepare sub projects for the various coders, do not put them into 'Inter-coder Mode'. This first needs to be done, when you merge the sub projects of the coders.

Summary of the steps

- The principal investigator (PI) develops the code system providing clear definitions for each code.
- The PI makes a copy of his or her project (see "Project Duplication") that either contains only documents and the code system, or documents, pre-defined quotations, and the code system (see "Removing Codings").
- The PI creates a project bundle file of the project for the coders. See "Exporting the Project for Distribution".
- The coders import the project bundle file. See "Importing a Project "
- The coders rename the project file after import. See "Renaming the Project after Import".
- The coders double-check under which user name they are logged in. See "User Accounts".
- The coders code the data independently. If they have questions or difficulties applying the codes, they write notes into a memo.
- Once the coders are done, they prepare a project bundle file for the PI. See "Exporting the Project for the PI".
- The PI imports the project bundle files from each coder and briefly checks the work of each coder and reads the memo they have written (if any).
- The principal investigator starts the inter-coder agreement mode and merges the projects of the coders. See "Merging Projects".
- The principal investigator runs the ICA analysis. See "Running the ICA Analysis".

Below you find technial instructions for both the principal investigator and the coders for what they need to know in terms of project management.

Project Management for the Principal Investigator (PI)

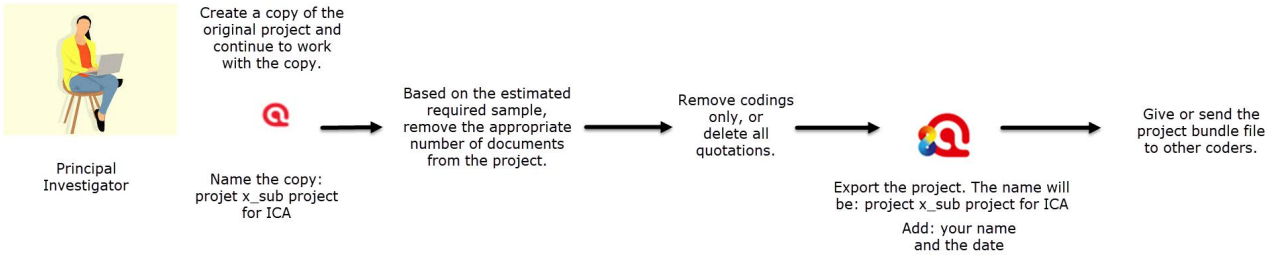

Figure 7: Project set-up for ICA analysis

Project Duplication

- Select **Project / Open.**
- Select a project and right click on the project name. Select the option: **Duplicate.**

The project that you want to duplicate must be closed.

Removing Codings

If you want to create a sub project for coders that contains pre-defined quotations:

- Select **Quotations / Remove Codings.**

Your project now contains documents, pre-defined quotations and codes. The codes show a frequency of 0. All codings has been removed from all quotations. Thus, all quotations are uncoded.

Exporting the Project for Distribution

- To export the project for distribution to all team member, select **Project / Export Project.**

Merging Projects

The Merge Tool reunites projects that were originally divided for analytical or economical reasons. It brings together the contributions of the coders. In the following a few general principles about merging are explained:

Master And Imported Projects: The Master project is the project into which another project, called the "Import project" is merged. The Master project has to be loaded first before invoking the **Merge Project** option.

Merge Strategy

The default option is to unify all objects that are **identical** and to add all objects that do not yet exist in the Master project. This is why it is so important to start from a common Master project, create sub projects from it and distribute those to the coders.

Identical Entities

When an entity is created in ATLAS.ti– regardless if it is a document, a code, a quotation, a memo, a network,a group or a comment--this entity receives a unique ID, comparable to a fingerprint. When merging projects, ATLAS.ti compares the IDs of the various entities. If they have the same ID (fingerprint), they are unified. If the fingerprint is different, they are added.

As coders do not add codes when serving as coders for an ICA analysis, there should never be duplicated codes!

Identical entities are unified, all others are added.

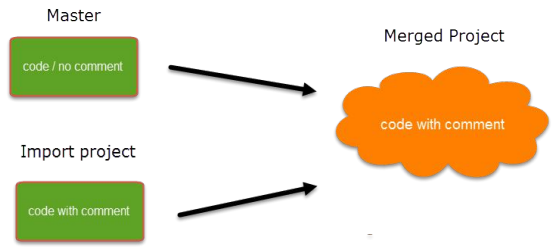

Figure 8: Merge example 2: Entities with comments, no conflict

Entities with and without comments: If there is a code C in the Master project that has no comment, and a code C in the Import project that has a comment, the merged procedure will add this comment to the merged project.

In the case of an unsolvable conflict (e.g., code C in the Master project has a comment, and code C in the Import project also has a comment) the user can define which of the two conflicting entities will win. If the comment of the Master project should be kept, you need to select the option "**Keep**." If the comment of the Import project should be kept, you need to select the option "**Override**".

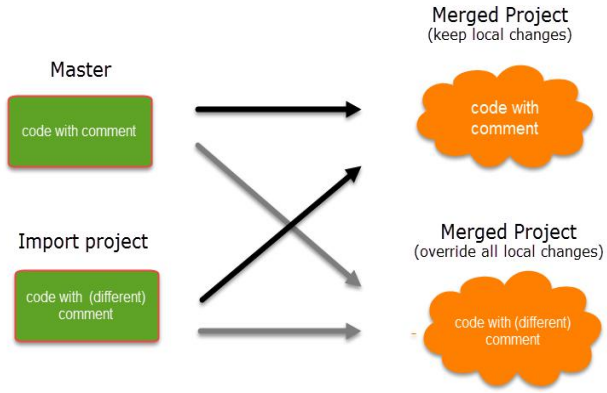

Figure 9: Merge example 3: Entities with conflicting comments

Coders should be instructed not to change code comments. If they want to write notes, they should create a memo and use it as writing space. **If you merge a project for ICA analysis, you should never encounter a merge conflict related to codes!**

How to Merge

- Open the project of coder 1.
- Set the project to ICA Mode by selecting **Project / Intercoder Agreement Mode**
- Merge the project of coder 2: **Project / Merge Project**. If there are more than 2 coders, continue to merge other projects.

In ICA mode, the codings of all coders are added, so you can compare them in the margin area and the ICA tool. If you hoover over a coding with your mouse in the margin area, you see the coder name. See Figure 10.

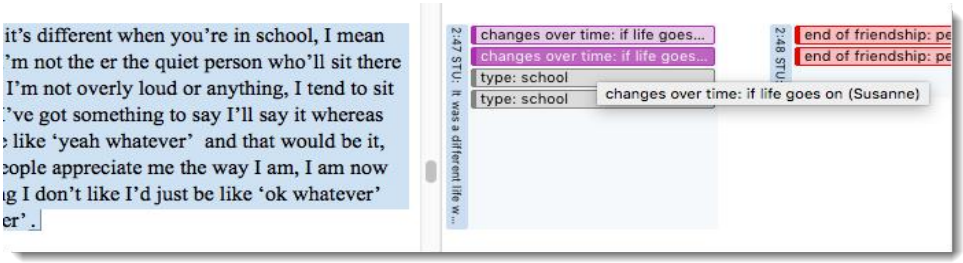

Figure 10: Display of the codings of different users

A coding is the link between a code an a quotation. In ICA mode, you can see which coder has applied which code to a quotation.

Project Management for Coders

User Accounts

ATLAS.ti automatically creates user names based on the account information on your computer.

- If you want to check under which name you are working, select the main menu option **Tools / User Management / change User.**
- If you want to create a new user account, select **Show User Management**

Importing a Project

You can open the project by double-clicking on the file, or select **Project / Import Project** from the main menu.

Renaming the Project after Import

After you imported the project, select **Project / Rename** rom the main menu and add your name or initials to the project name.

Exporting the Project for the PI

After the coding is done, as coder you need to create a project bundle file and send it back to the principal investigator.

Select **Project / Export Project.** Save the file to your hard disk, an external or a cloud drive.

It is recommend that all team members save a copy of the project bundle file on their computer, or their personal space on a server.

Running the ICA Analysis

After the projects are merged, the ICA analysis can be conducted:

To start the ICA Analysis, select **Analysis / Calculate intercoder Agreement**.

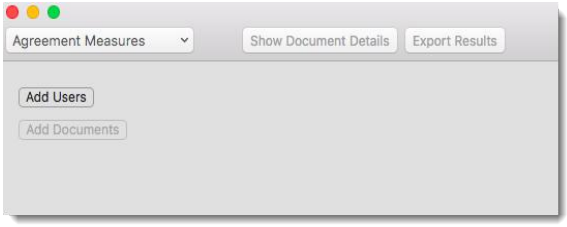

Figure 11: Opening screen of the ICA tool

Click on the **Add User** button and select two or more coders.

Click on the **Add Documents** button and select the documents that should be included in the analysis. The default option is for all selected documents to be regarded as one continuum and all coded quotations of these documents will go into the calculation. You can, however, also view agreements / disagreements per document and get a coefficient for each document.

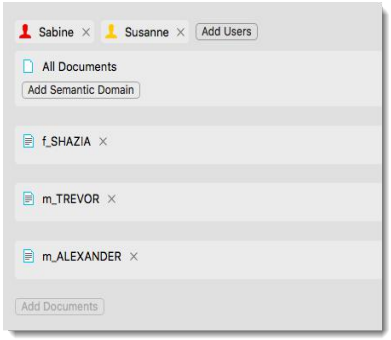

Figure 12: Adding codes, documents and codes for the ICA analysis

Click on the **Add Semantic Domain** button and add one or more codes to the domain. See the chapter on "Semantic Domains"for further information.

You can also add single codes to semantic domains. You can then however not calculate the cu-alpha coefficient as it is specific to semantic domains. See also "Methods for Testing ICA".

Repeat this process for each semantic domain.

As there currently is no function to define a semantic domain formerly, all codes that you are adding from the list will be considered to belong to a semantic domain. As you can see in Figure 13, semantic domains were already built indirectly in the process of developing the code system by adding the same prefix to all codes that belong to the same semantic domain like ACTIVITIES. Please remember that all sub codes of a semantic domain (i. e., all codes with prefixes) must be applied to quotations in a mutual exclusive manner. See "Rules for Applying Codes".

After you have added coders, documents and codes, your screen looks similar to what is shown in Figure 13.

The quotations of the semantic domain are displayed on the right-hand side. When you select a quotation, the full content is shown in the preview and you can write comments. If you double-click, you can view the quotation in context.

If you click on one of the codes, only the quotations of the selected code and coder are shown in the quotation window:

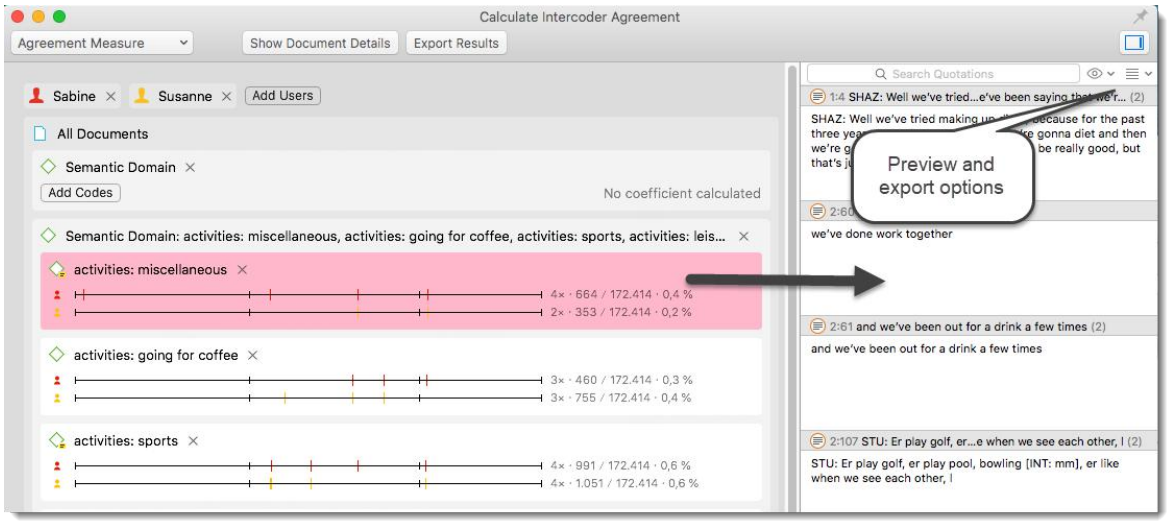

Figure 13: Review quotations for each code and coder

Currently, it is only possible to display the quotations of both coders, and not separately by coders. Also there is an issue with the report option for quotations – the report contains ALL quotations and not only those for the selected code.

You can remove coders, codes or documents from the analysis by clicking on the X.

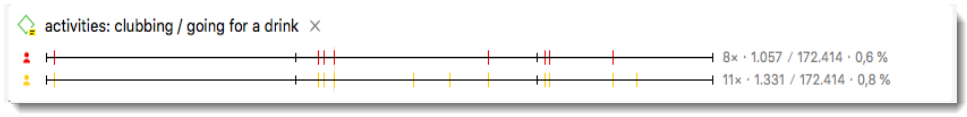

Figure 14: Descriptive statistics

To the right of each coder line you see some descriptive statistics:

The red coder has applied the code "activities clubbing / going for a drink" 8 times. The number of characters that are coded with this code are 1057 out of a total of 172.414 characters in all three selected documents, which is 0,6%.

The yellow coder has applied the code "activities clubbing / going for a drink" 11 times. The number of characters that are coded with this code are 1331 out of a total of 172.414 characters in all three selected documents, which is 0,8%.

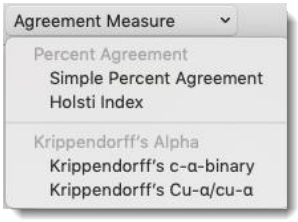

Figure 15: ICA Agreement measures

Calculating an ICA Coefficient

To calculate a coefficient, select one of the four method. See "Methods for Testing ICA" for more detail. Below the results of the Krippendorff's coefficients are explained in more detail.

Krippendorff's Alpha

The c-alpha binary for the semantic domain 'activities' in Figure 16 is 0,896. This means that the reliability for identifying relevant from irrelevant matter related to the codes of this domain is high.

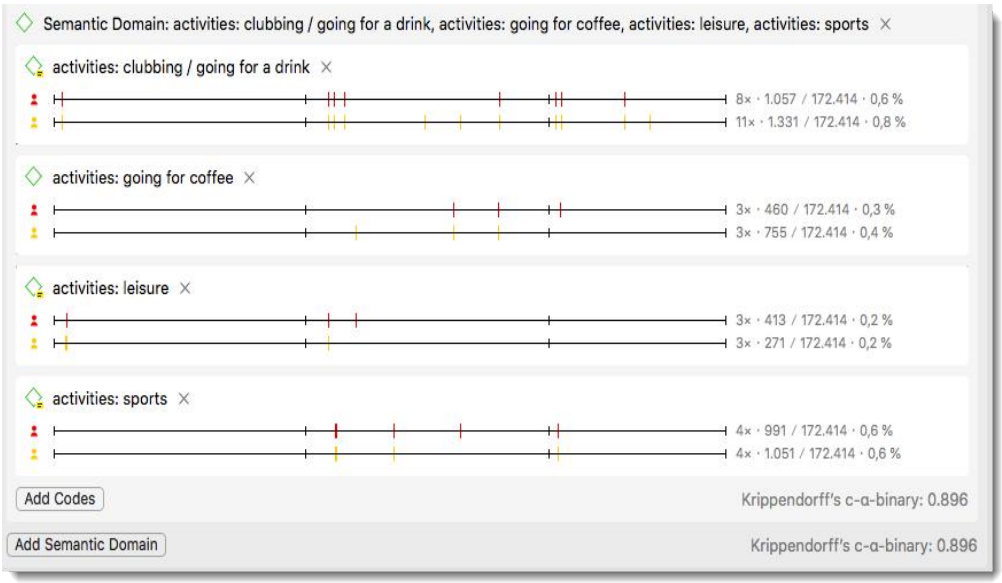

Figure 16: Display of results for c-alpha binary

The results for cu-alpha and Cu-alpha for the two selected domains are shown in Figure 17.

If multiple sub codes of the same semantic domain have been applied to a quotation, this is a violation of the mutual exclusive coding rule and Krippendorff's cu-alpha cannot be computed.

Figure 17: Display of results for cu-alpha and Cu-alpha

The difference between the two measures is that the lower case **cu-alpha** applies to one semantic domain only, the capital **Cu-alpha** applies to all selected semantic domains. It is not simply an average of all lower case cu-alphas, as sub codes from different domains can potentially be applied to the same quotations. This is referred to as multi-valued coding (see Krippendorff et al., 2016) and the section on "Rules for Applying Codes" in this manual.

Percent Agreement is likely to be higher than Krippendorff's alpha as it does not take into acount change agreement. This means you overestimate true agreement.

Exporting Results

Export Results

The results of an ICA analysis can be exported to Excel. See **Export Results** button on top of the screen. The reports looks as follows:

The report contains the following:

First line: project name

Second line: Type of coefficient, if one was selected.

Legend: Explains the column headers of the table

Followed by **coders** and **documents** used in the analysis

Above each table the **semantic domain** and its codes are listed that go into the analysis.

The table contains the **descriptive statistics** for each code and coder, and if a **coefficient** was selected, the resulting number.

If you have chosen the detailed view for each document in the ICA tool, the table contains the analysis for all documents plus the analysis for each individual document.

Intercoder Agreement for Project:	friendship project_Master (coder Susanne und Sabine_ICA Test)				
Agreement Coefficient:	Krippendorff's Cu-Alpha / cu-Alpha				
Legend					
Applied*	number of times the code has been applied				
Units*	number of units* the code has been applied				
Total Units*	total number of units* across all selected documents				
Total Coverage*	% coverage within the selected documents				
Coders					
	Sabine				
	Susanne				
Documents					
	f_SHAZIA				
	m_TREVOR				
	m_ALEXANDER				
All Documents					
Semantic Domain:	activities: clubbing / going for a drink, activities: going for coffee, activities: leisure, activities				
Code	Coder	Applied*	Units*	Total Units*	Total Coverage*
activities: clubbing / going for a drink					
	Sabine	8	1057	172414	0,61%
	Susanne	11	1331	172414	0,77%
activities: going for coffee					
	Sabine	3	460	172414	0,27%
	Susanne	3	755	172414	0,44%
activities: leisure					
	Sabine	3	413	172414	0,24%
	Susanne	3	271	172414	0,16%
activities: sports					
	Sabine	4	991	172414	0,57%
	Susanne	4	1051	172414	0,61%
			Reliability Coefficient		
			Krippendorff's Cu-Alpha / cu-Alpha: 0.833		

Figure 18: ICA Excel report

References

Cohen, Jacob (1960). *A coefficient of agreement for nomincal scales.* Educational and Psychological Measurement, 20, 37–46.

Guba, Egon G. and Lincoln, Yvonna S. (2005). *Competing paradigms in qualitative research*, in Denzin, N. and Lincoln, Y.S. (eds) The Sage Handbook of Qualitative Research, 191–225. London: Sage.

Harris, Judith, Pryor, Jeffry and Adams, Sharon (2006). The Challenge of Intercoder Agreement in Qualitative Inquiry. Working paper.

Holsti, O. R. (1969). *Content Analysis for the Social Sciences and Humanities*. Reading, MA: Addison-Wesley

Krippendorff, Klaus (2004/2019). *Content Analysis: An Introduction to Its Methodology (3rd and* 4th edition). Thousand Oaks, CA: Sage.

MacPhail, Catherine, Khoza, Nomhle, Abler, Laurie and Ranganathan, Meghna (2015). P*rocess guidelines for establishing Intercoder Reliability in qualitative studies*. Qualitative Research, 16 (2), 198-212.

Mouter, Nick, Vonk Noordegraaf, Diana (2012). Intercoder reliability for qualitative research. You win some, but do you lose some as well? TRAIL Research School, Faculty of Technology, Policy and Management, TU Delft, Netherlands.

Rolfe, Gary (2006). Validity, trustworthiness and rigour: quality and the idea of qualitative research. Methodological Issues in Nursing Research, 304-310.

Richards, Lyn (2009). *Handling Qualitative Data: A Practical Guide*, 2nd edn. London: Sage.

Schreier, Margrit (2012). *Qualitative Content Analysis in Practice*. London: Sage.

Zwick, Rebecca (1988). *Another look at interrater agreement*. Psychological Bulletin, 103, 347–87.