**Testing intra- and inter-coder agreement**

**How many (and which) units to include in a coding agreement test**

There are no strict rules regarding the number of units that need to be included in the coding agreement test. The rule of thumb you sometimes read (defining a percentage of the total sample to determine the number of units for the coding agreement test) should not be used. The main impact of selecting more units are the following:

* **You get more variation on the variables.** If you have too few units and conduct tests for many variables, you risk to have the same “true” result for a variable across all units. If e.g. you want to code how often the health minister appears in frontpage news and you include too few articles, you run the risk that the health minister does not appear a single time. Based on this, you cannot estimate whether coders accurately recognize if the health minister appears or not.
* **You get slimmer confidence intervals.** When you calculate coding agreement indices (see below), you have to reckon that these are only estimates with some uncertainty. This is expressed by confidence intervals: with a relatively high probability (e.g. 95%), the real value lies somewhere in an interval around the estimated value (that is a 95% confidence interval). The more units you include in the test, the slimmer does the interval get and the estimate becomes a more reliable approximation of the real value.

If you can afford to code 100 or more units several times by all coders, that is clearly good. Less than 30 units may cause problematically low variation on some variables, and induce too much uncertainty into the results (i.e. too wide confidence intervals).

It should be considered that having more than two coders also leads to more comparisons and thereby slims the confidence intervals. Therefore a high number of coders involved in the agreement task can justify reducing the number of units.

For selecting which units should go into the coding agreement test, a random selection is generally a good solution. However, it can make sense to sample specific units to increase the extent of variation on variables with low variation. In the case of the health minister, we could actively seek out articles that feature the health minister such that the test will be more useful to assess the coders’ performance on that variable.

**How does a coding agreement test (and the data it generates) look like and how does it work?**

We conduct a content analysis of frontpage newspaper articles in Aftenposten with four different categories. Each category exemplifies four different types of scaling that are commonplace in content analysis:

* Category 1: Does the article feature the Norwegian prime minister (=1) or not (=0)?
* Category 2: How many words does the article feature (a count variable).
* Category 3: Is the article’s sentiment towards the Norwegian prime minister exclusively negative (1), mostly negative [but also with some positivity] (2), ambivalent [positive and negative in equal proportions] (3), mostly positive [but also with some negativity] (4), or exclusively positive (5) (an ordinal variable)
* Category 4: Is the article’s main topic (a) politics, (b) economy, (c) sports, or (d) anything else?

Three coders are involved in the coding process and are included in the agreement test.

We choose 8 articles that are to be analyzed regarding these four categories. Often you would choose for units for the agreement test, but lower numbers of units make the examples easier to understand. All three coders code all 8 articles such that we can assess their agreement in direct comparison regarding the same articles.

For an intra-coder reliability test, one coder would repeat coding the articles at a later time (e.g. four weeks after having completed the first round of coding). Otherwise, the procedure is the same as for an inter-coder agreement test.

The tables below provide example results for the four variables. Other than regular data sets, this data set can be thought of as three-dimensional (rather than two-dimensional): case and variable are the typical formats for organizing data. For agreement data, the coder (or sometimes called: rater)

This involves some math and we should clarify the use of signs and indices:

The ten units are called U. Capital “U” stands for all units, u1 stands for the first unit, u2 for the second unit, and so forth (uu for the u-th unit).

The three coders are called C. Capital “C” stands for all coders, c1 stands for coder 1, c2 for coder 2, and so forth (cc for the c-th coder)

The four variables are called V. Capital “V” stands for all variables, v1 stands for variable 1 (category 1), v2 for variable 2 (category 2), and so forth (vv for the v-th variable).

**Coding agreement data for variable 1 (“Prime minister mentioned?”)**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Data for v1 | u1 | u2 | u3 | u4 | u5 | u6 | u7 | u8 |
| C1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 |
| C2 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| C3 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 |

**Coding agreement data for variable 2 (“Word count”)**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Data for v2 | u1 | u2 | u3 | u4 | u5 | u6 | u7 | u8 |
| C1 | 158 | 44 | 231 | 117 | 301 | 125 | 80 | 46 |
| C2 | 162 | 44 | 234 | 118 | 301 | 128 | 78 | 48 |
| C3 | 177 | 58 | 254 | 135 | 324 | 145 | 97 | 61 |

**Coding agreement data for variable 3 (“Sentiment towards Prime minister?”)**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Data for v3 | u1 | u2 | u3 | u4 | u5 | u6 | u7 | u8 |
| C1 | 3 | 3 | 3 | 3 | 4 | 5 | 1 | 2 |
| C2 | 3 | 3 | 4 | 2 | 4 | 5 | 1 | 1 |
| C3 | 3 | 3 | 3 | 3 | 5 | 5 | 1 | 3 |

**Coding agreement data for variable 4 (“Main topic?”)**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Data for v3 | u1 | u2 | u3 | u4 | u5 | u6 | u7 | u8 |
| C1 | 1 | 1 | 2 | 3 | 4 | 1 | 1 | 2 |
| C2 | 1 | 2 | 2 | 3 | 4 | 4 | 1 | 4 |
| C3 | 1 | 1 | 2 | 1 | 1 | 1 | 1 | 2 |

**How to calculate coding agreement indices from the coding agreement data?**

The observed coincidence matrix for v1 (only c1 and c2) looks like this. See that the coincidence matrix is always symmetric and each potential coding agreement is counted twice, once for each of the two coders that are compared. The number of entries is calculated according to the formula: . In our case with three coders (c1, c2, c3) and eight units, the number of coincidences would be . The diagonal entries (grey cells) are agreements, the non-diagonal entries (white cells) are disagreements

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | C1 | | *Marginal sums* |
|  |  | No mention | Some mention |
| C2 | No mention of prime minister | 10 | 1 | *11* |
| Some mention of prime minister | 1 | 4 | *5* |
| *Marginal sums* | | *11* | *5* | *16* |

Calculating the coding agreement indices involves four components: (1) the **observed coincidence matrix** (a way to put the agreement data into a form that is more convenient for the computation), (2) the **expected coincidence matrix** (a theoretical coincidence matrix that describes what kinds of agreements would be expected by chance), (3) the **weighting matrix** (describing how “severe” disagreements that occur should be “punished” in the coding agreement index).

While the observed coincidence matrix is the same for all coding agreement indices, they vary in how they calculate the expected coincidence matrix (i.e. how they conceptualize/define chance agreement) and in the weighting matrix they use (but basically all agreement indices can be coupled with all weighting matrices – the choice of expected coincidence matrix and weighting matrix is independent.

Percent agreement: No expected coincidence matrix is calculated. Therefore, 0 means “no agreement” and negative values are not possible. The formula is simple:

or 87.5%.

**Krippendorff’s alpha:** The expected coincidence matrix is calculated from the joint marginal distribution across coders.[[1]](#footnote-1) 0 means “observed agreement is as high as chance agreement” and hence all observed agreement *might* be attributable to chance.

***Expected coincidence matrix for Krippendorff’s Alpha/Fleiss Kappa, based on observed marginal sums***

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | C1 | |  |
|  |  | No mention | Some mention | *Marginal sums* |
| C2 | No mention of prime minister | 11\*11/16=7.6 | 5\*11/16=3.4 | *11* |
| Some mention of prime minister | 5\*11/16=3.4 | 5\*5/16=1.6 | *5* |
| *Marginal sums* | | *11* | *5* | *16* |

or 70.9% agreement above chance.

**Brennan and Prediger’s kappa.** The expected coincidence matrix is calculated from the number of scale points, and all coincidences are distributed equally across the whole scale.

***Expected coincidence matrix for Brennan and Prediger’s Kappa, based on equal distribution across all scale points***

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | C1 | | *A priori marginal distribution* |
|  |  | No mention | Some mention |
| C2 | No mention of prime minister | 8\*8/16=4 | 8\*8/16=4 | *8* |
| Some mention of prime minister | 8\*8/16=4 | 8\*8/16=4 | *8* |
| *A priori marginal distribution* | | *8* | *8* | *16* |

or 75.0% agreement above chance.

**The weighting matrix:**

When we look at the coding of word numbers per article, the above formulas would suggest very low agreement. And in fact coders 1 and 2 rarely agree perfectly. Our intuition, however, would say that the differences by one or two words are marginal – and that intuition is right; this suggests that there is some measurement error, but that it is not very bad.

This problem can be solved by using an appropriate weighting matrix. Both the observed and the expected coincidence matrix is multiplied with that weighting matrix. The weighting matrix states how bad a disagreement actually is compared relative to other disagreements.

The expected disagreement is harder to calculate but if we compare how much of a penalty randomly mixing the coders’ assigned word counts produces (as an approximation of chance agreement), we observe that the observed disagreement score (“weighted sum”) is dwarfed by the expected disagreement score for linear and quadratic weighting matrices.

**Coding agreement data for variable 2 (“Word count”) for the raw data**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Data for v2 | u1 | u2 | u3 | u4 | u5 | u6 | u7 | u8 | Weighted sum |
| C1 | 158 | 44 | 231 | 117 | 301 | 125 | 80 | 46 | — |
| C2 | 162 | 44 | 234 | 118 | 301 | 128 | 78 | 48 | — |
| **Penalty** | | | | | | | | | |
| Identity weighting matrix | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 6 |
| Linear weighting matrix | 4 | 0 | 3 | 1 | 0 | 3 | 2 | 2 | 15 |
| Quadratic weighting matrix | 4\*4 | 0\*0 | 3\*3 | 1\*1 | 0\*0 | 3\*3 | 2\*2 | 2\*2 | 43 |

**Coding agreement data for variable 2 (“Word count”) after shuffling the data (as an approximation of chance agreement)**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Data for v2 | u1 | u2 | u3 | u4 | u5 | u6 | u7 | u8 | Weighted sum |
| C1 | 158 | 44 | 231 | 117 | 301 | 125 | 80 | 46 | — |
| C2 | 301 | 128 | 78 | 48 | 162 | 44 | 234 | 118 | — |
| **Penalty** | | | | | | | | | |
| Identity matrix | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 8 |
| Linear matrix | 143 | 84 | 153 | 69 | 139 | 81 | 154 | 72 | 895 |
| Quadratic matrix | 20449 | 7056 | 23409 | 4761 | 19321 | 6561 | 23716 | 5184 | 110457 |

**Code in R**

Load or enter the data. If you enter the data in an Excel spreadsheet (one per coder, variables as columns (called “v1”, “v2” etc. in the first row) and units as rows as in a regular data set) and save it as “c1.csv”, “c2.csv”, etc., start R and execute the code:

c1 <- data.frame(read.csv2(“c1.csv”))

c2 <- data.frame(read.csv2(“c1.csv”))

c3 <- data.frame(read.csv2(“c1.csv”))

For variables 1, 2, … (v1-v4), you can create the raw data matrix like this:

v1 <- cbind(c1$v1,c2$v1,c3$v1)

v2 <- cbind(c1$v2,c2$v2,c3$v2)

v3 <- cbind(c1$v3,c2$v3,c3$v3)

v4 <- cbind(c1$v4,c2$v4,c3$v4)

Then you have to get the appropriate packages for calculating coding agreement:

install.packages(“irrCAC”) # only the first time you install this package; afterwards you can just load the library.

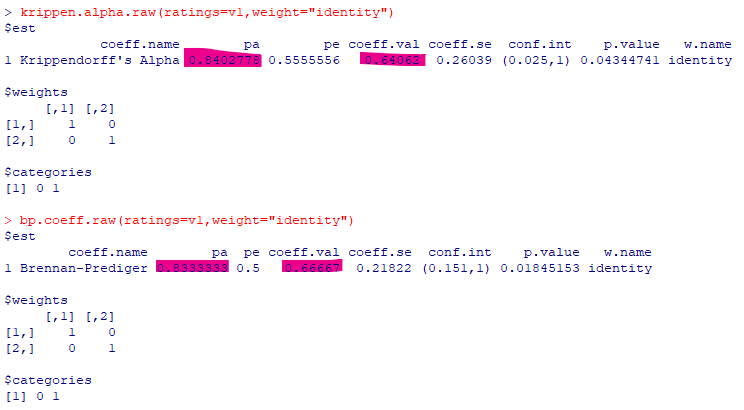
library(irrCAC)

The commands are: krippen.alpha.raw() for Krippendorff’s alpha and bp.coeff.raw() for Brennan and Prediger’s kappa. Both will also print results for percent agreement, so no specific function for percent agreement is needed. Both functions work in a similar way: first you supply the data (v1, for example), and then you specify the weighting matrix to be used. “Identity” means that you are using categorical/nominal scaling; “ordinal” that you are using a rank-order/ordinal scale, “linear” and “quadratic” can be used for metric scales (conventionally, “quadratic” is preferred, weighting severe disagreements particularly strongly).

krippen.alpha.raw(ratings=v1,weight=”identity”)

bp.coeff.raw(ratings=v1,weight=”identity”)

**Figure 1:** Output from using irrCAC package in R



If you try it with v2, this shows how important it is to use an appropriate weighting matrix (in this case: a “quadratic” weighting matrix).

krippen.alpha.raw(ratings=v2,weight=”identity”) # .07664, barely above chance

bp.coeff.raw(ratings=v2,weight=”identity”) # .03968, barely above chance

krippen.alpha.raw(ratings=v2,weight=”quadratic”) # .9855, almost perfect

bp.coeff.raw(ratings=v2,weight=”quadratic”) # .98274, almost perfect

**How to interpret the coefficients**

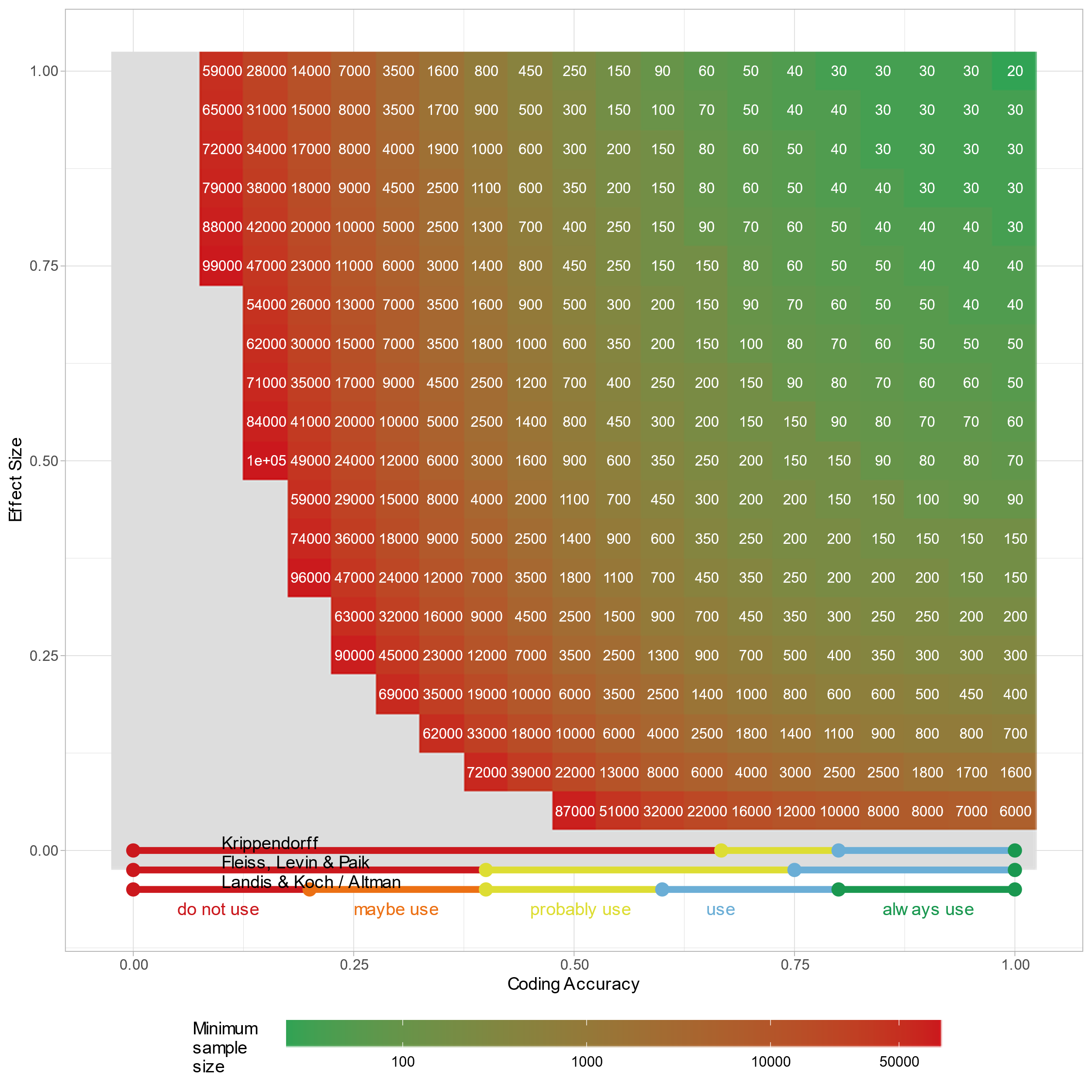
There are different rules of thumb for what coefficients indicate sufficient levels of agreement. The most widely used is Krippendorff’s recommendation for Krippendorff’s alpha (that can also be applied to Cohen’s kappa, Fleiss’ kappa because of the many similarities they share; with some caution it can also be used for Brennan and Prediger’s kappa): values >0.800 generally indicate that the coding can be used. Values <0.667 indicate that the coding should not be used. Values between .667 and .800 can be used with caution, mostly for exploratory research.

These rules of thumb are rough and do not fit in every situation. A simulation study (Geiß, 2021) noted that besides coder agreement, sample size (not of the coding agreement task, but of the final study) and the effect size of the relationship that is investigated must be taken into consideration. The effect size is not known a priori, but can be estimated based on previous research or just assumed to take a moderate value (e.g. r=0.3).

These calculations suggest that in many settings the lower threshold defined by Krippendorff is sufficient to have sufficient statistical power in hypothesis tests.

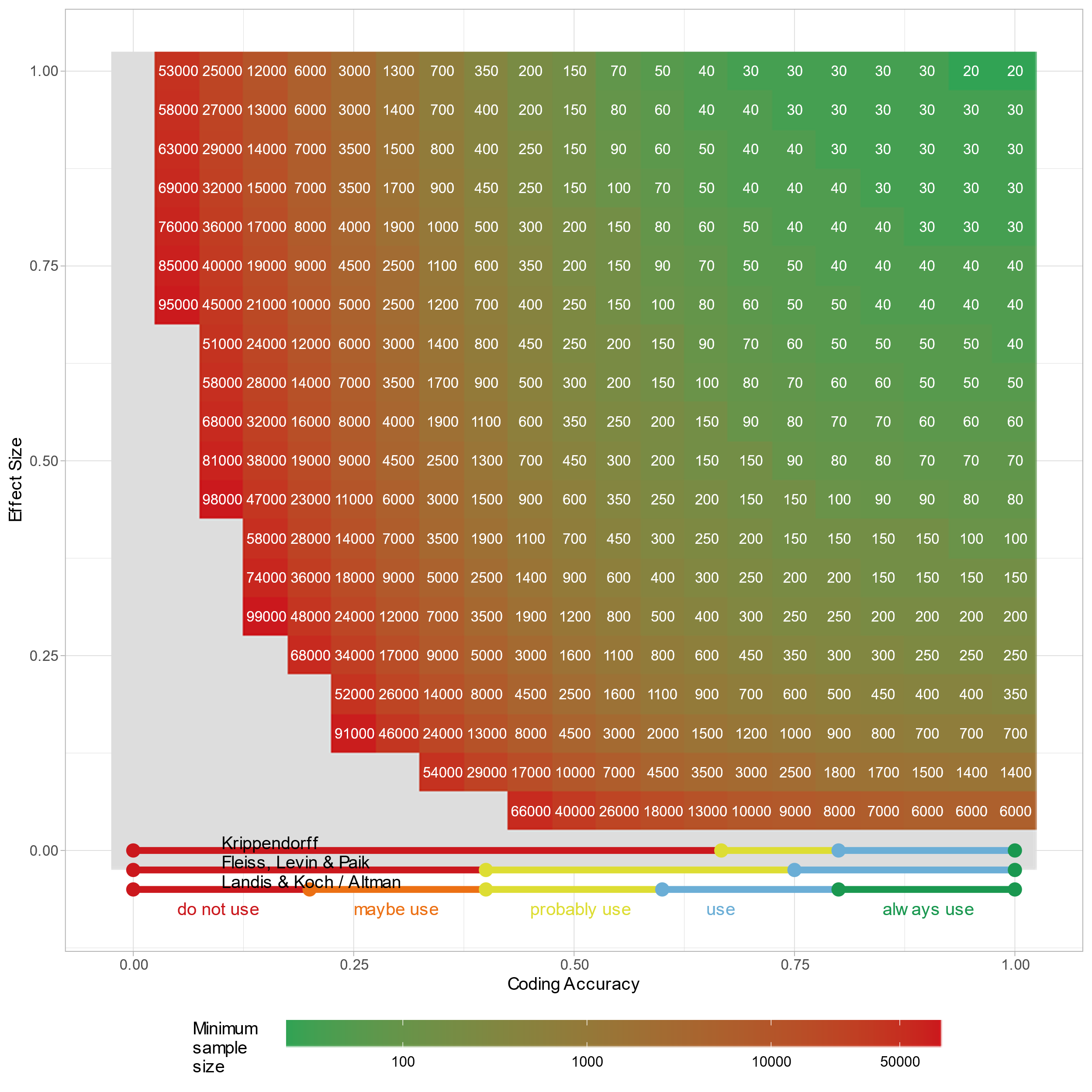
You can also use these graphics to design your study in terms of the sample sizes you are using.

**Figure 2:** Sample size, coding accuracy (estimated by coding agreement) and effect size (R) if coders guess based on the scale (as e.g. assumed by Brennan and Prediger’s Kappa).

**

Reading example: If you expect an effect size of 0.25, and obtain a coding agreement coefficient of 0.50, you need a sample size of 3500 to have sufficient statistical power for testing a hypothesis (lower than 5% alpha error probability and lower than 5% beta error probability)

**Figure 3:** Sample size, coding accuracy (estimated by coding agreement) and effect size (R) if coders guess based on the scale (as e.g. assumed by Krippendorff’s Alpha)

**

Reading example: If you expect an effect size of 0.25, and obtain a coding agreement coefficient of 0.50, you need a sample size of 1600 to have sufficient statistical power for testing a hypothesis (lower than 5% alpha error probability and lower than 5% beta error probability)

**What to do if coding agreement is insufficient**

If coding agreement is insufficient, there are several options what you can do. You should always test for sufficient agreement before starting the actual coding such that you do not have to scrap the results you have obtained at the time the coding agreement test shows that the coding is not reliable enough.

* Check if you have chosen an appropriate index of coding agreement.
* Check if you have chosen an appropriate weighting matrix (according to the scaling of the variable)
* Check if considering sample size and expected effect size can help you. Maybe you can increase sample size if coding agreement is only marginally below the rule-of-thumb standard
* Check if you have sufficient variation on the variables that have a low agreement index; often, the cause is that there are no or too few positive (or too few negative) or the variation does not span the entire scale that is used. For instance, you need articles of different lengths to conduct a robust test of coding agreement.
* Look at the coding agreement data and see whether it is just one coder that always deviates from the others. That coder may hold misconceptions, or the coding instructions are not entirely clear to that person. You can either train that coder again or remove him/her from the coder team.
* Get feedback from your coders what they think about the categories that yield problematic agreement. Did they experience much uncertainty while coding? Do they have any ideas how the coding instructions could be refined or simplified?
* Take a look at the units in which severe disagreements occurred. Why is that?
* Based on all this information, revise the codebook and update the coder training. Use more examples in the coder training and let coders discuss their disagreements and try to find a common standard. Once that standard is established, try to include it into the coding instructions such that the standard is reproducible.

**What NOT to do if coding agreement is insufficient**

* Do not play around (rescale, recategorize, remove outliers, etc.) until you get a sufficiently high value. Would you do all these checks with the real data? If the goal is just to increase the agreement index value (and not to make the test more appropriate), then leave it be. At least, you should document what you did to the data and explain why.
* Do not run several tests without making real changes to the coding procedures or the coder training. It is very unlikely that you just had “bad luck”.
* Do not run the next test with the same material after telling the coders the “right” solution. They will remember it and the coding agreement will be artificially high.
* Do not make fundamental changes or standardizations in coder training that are not documented in the code book. Document any major changes of instructions, any heuristics that coders agree on using in the code book.

1. Krippendorff’s Alpha also includes a slight adjustment based on the number of units (U). If this adjustment is not done, it is called Fleiss kappa. The differences between Fleiss’ kappa and Krippendorff’s alpha is marginal and grows smaller the greater U is. [↑](#footnote-ref-1)