***Compare partitions***

SPSS macros by Kirill Orlov

kior@akado.ru, ttnphns@gmail.com

<https://www.spsstools.net/en/KO-spssmacros>

All rights reserved

*Comparison of classifications/clusterings.* Computation of various measures of likeness of groupings: external clustering criteria, classification performance and agreement indices. Identification of same or similar groups between groupings.

*Read “*[*About SPSS macros*](https://www.spsstools.net/en/KO-aboutmacros)*” what are they and how to run them.*

*The “Protected directory” error.* Some of the macros described in the current document write temporary files to hard disc. If you don't have full Administrator rights of your computer, it may cause error saying, among things: *“SPSS Statistics cannot access a file... specifies a protected directory...”*, meaning that the default directory the macro wants to use is protected on your PC. To solve the problem, in Syntax window issue command: CD 'myfolder'., where 'myfolder' is the path/name of some folder where you are allowed to save files to.

Suppose you want to obtain an index of argeement (similarity) between two partitions *U* and *V* of the same set of objects into (nonintersecting) groups. Partitions *U* and *V* are alike when in both partitions there are the same, by object composition, groups. Before comparing the partitions there arises the question: ***do we know in advance which group from U which namely group from V corresponds to, or we don’t know?*** Apparently, if *U* and *V* are “labeled” (identified) classes, we (1) know beforehand to which one class from *U* which one class from *V* corresponds. But if *U* and *V* are “unlabeled” clusters or one of the two partitions is classes but the other is clusters, then the question emerges, do we need to previously (2) find such one-to-one group matches between the partitions or (3) we can do without the explicit one-to-one pairing of groups?

* Variant (3) is answered by macro [!KO\_CLUAGREE](#_МАКРОС_!PROXQNT:_РАЗНЫЕ) for calculating similarity measures between “unlabeled” partitions (i.e. group matching between partitions is not established). This macro computes proximity measures that either don’t rely on group correspondence between partitions *U* and *V*, or correspondence between their groups is somehow determined implicitly by the very formula of the proximity measure. Typical usage example is a comparison of clusterings or of a clustering with a classification.
* Variant (1) is answered by macro [!KO\_CLASAGREE](#_МАКРОС_!CLASAGREE:_МЕРЫ) for calculating similarity measures between “labeled” partitions (i.e. group matching between partitions is established) . This macro computes proximity measures that are based on information whether an object is “correctly” or “incorrectly” classified in partition *V* relative partition *U*. Typical usage example is a comparison of classifications.
* Variant (2) is answered by macro [!KO\_GRMATCH](#_MACRO_!GRMATCH:_IDENTIFICATION) for establishing of mutually correspondent groups between partitions. If you need you can, after recoding of groups by this macro, then use macro !KO\_CLASAGREE.

# MACRO !KO\_CLUAGREE: COMPARISON MEASURES FOR “CLUSTERING” PARTITIONS

Version 1, Mar 2019. Tested on SPSS Statistics 17, 20, 22.

!KO\_cluagree vars= v1 v2 v3 v4 /\*Variables representing case groupings;

/\*word WITH is possible after the first name

/measure= DICE /\*Specify measure(s): DICE, OCHIAI, KULCZ2, JACCARD, RAND, ARAND, RR, RT,

/\*SS2, SS1, PHI, DISPER, SEUCLID, MN, OVL, FCA, MI, NMI, VI, AMI, HCV

/asymm= /\*For asymmetric measures: don’t return asymmetric version (NO, default),

/\*return both asymm values (BOTH), return UV, return VU

/normz= /\*Normalization for NMI, AMI: MAX (default), MIN, MEAN, GMEAN

/print= /\*Show technical details: NO (default) or YES.

Minimal specification VARS, MEASURE.

The macro computes various measures of agreement between partitions, known as *external clustering criteria*, *supervised cluster validity indices*, and other names. The point is to compare two partitions and return a measure of their similarity. Two partitions are similar, close, if they consist of almost the same groups of objects, that is, when it is true for many of the objects that objects forming a group in one partition mostly fall together in a group in the other partition. The macro calculates quantitative measures of proximity of partitions. Word “partition” is the synonym of g-class grouping (nonoverlapping classes).

In contrast to [!KO\_CLASAGREE](#_МАКРОС_!CLASAGREE:_МЕРЫ) macro, this macro **does not require that there is known a namely correspondence between groups of the being compared partitions**. In other words, “group labels” are not required. That means that the group coding can be arbitrary and different in different partitions; codes don’t play a role. Usually this way there are compared results of *clusterings* or results of a *clustering* with a *classification* or a reference grouping, compare of *clustering* with *external* (background) grouping to validate the clustering. Generally, partitions may be results of any grouping by origin, while, as said before, a specific equivalence of groups between groupings is not set.

Objects are dataset cases and partitions are variables. There can be two or more partitions, the macro compares them in twos. Partitions must be mutually exclusive groups ( = alternative groups, crisp groups), i.e. each object belonging to precisely one group (a group may consist even of one object). This macro does not produce measures for fuzzy groupings. The macro makes no use of any data features based on which the classifications/clusterings were obtained. Only object groupings is the information input to the analysis.

The macro outputs the computed measures in Viewer and in new dataset. See detail in s/c VARS.

EXAMPLE 1.

!KO\_cluagree vars= true WITH clu1 clu2 clu3 /measure= RAND ARAND MI AMI.

* Three different cluster solutions (got by different clustering methods), CLU1, CLU2, CLU3, are compared each with the standard object partition, TRUE, by indices: Rand, Adjusted Rand, Mutual Information, Adjusted Mutual Information.

EXAMPLE 2.

!KO\_cluagree vars= clu1 to clu10 /measure= FCA.

* Results of different clusterings (different methods, various number of clusters), variables CLU1 through CLU10, are compared with each other in pairs by F Clusteting Accuracy similarity index, and the square matrix is output.

***Subcommands***

**VARS**

Specify input numeric categorical, nominal variables (names up to 8 bytes long) which are the being compared partitions of the objects (the dataset cases). Each unique value in a variable signifies a group. At least two variables; you can use word “to” in the list for contiguous variables in the dataset. The variables must have no missings (if missings, cope with them in advance: you could decide to exclude objects with missings or enlist them in a single new group or each such object in its own group). Each variable must be not a constant and must have at least one duplicating value (that is, the number of groups must be greater than one and less than the number of objects). The macro does not check the correctness of the input variables, so you should watch yourself.

There are two modes to specify VARS: with keyword WITH and without it. If without WITH, for instance:

VARS= *CLU1 CLU3* to *CLU10*

Then you have the right to request only one proximity measure in s/c MEASURE. The macro will compute that proximity measure between all partitions VARS pairwisely and will return, as a new dataset, the square matrix with the proximity values. Each partition of the list once comes out as partition *U*, the other time as partition *V*, when they are compared.

If to use word WITH, then it must go *second*, after the name of one partition that we denote *U*. Every partition following the word WITH, we denote *V*. So, one specified *U* will be compared with each of *V*. For example:

VARS= *TRUECLAS* WITH *CLU1 CLU3* to *CLU10*

Partition *TRUECLAS* will be compared with each listed after WITH. This is the comparison of one partition with all the rest. You can request more than one proximity measure in s/c MEASURE. The macro will return the rectangular result matrix in a new dataset and in Viewer window.

**ASYMM**

This subcommand applies to measures that can be asymmetric[[1]](#footnote-1) (asymmetric in the macro can be the following measures: DICE, OCHIAI, KULCZ2, OVL, FCA, HCV; specifically for each of them see s/c MEASURE), i.e. where two values are computable at comparison of *U* and *V* partitions, *MeasureUV ≠ MeasureVU*. Measures that are computed only as symmetric (*MeasureUV = MeasureVU*) ignore the subcommand.

NO - (also default/unspecification) return symmetric version of the measure that can be asymmetric.

UV - return value *MeasureUV*.

VU - return value *MeasureVU*.

BOTH - return both values, *MeasureUV* (in the above-diagonal triangle of the square matrix, in cell *UV*) and *MeasureVU* (in the below-diagonal triangle of the square matrix, in cell *VU*). This option is incompatible with keyword WITH in VARS.

**MEASURE**

Select a proximity measure or measures (in the presence of WITH in VARS list) from the list. The macro computes two types of measures.

**A)** Measures based on **comembership confusion matrix**, or **object pairs** measures. Let there be *N* objects and two being compared partitions into groups, partition *U* and partition *V*. All possible *pairs* of objects get considered and the following 2x2 frequency table (comembership confusion matrix ) counting pairs is filled in:

|  |  |  |  |
| --- | --- | --- | --- |
|  | *V* | |  |
| *U* | *a* | *b* | *a+b* |
| *c* | *d* | *c+d* |
|  | *a+c* | *b+d* | *a+d+b+c = N(N-*1*)/*2 |

according to rule:

* + If a pair is found in one group in *U* and is found:
    - * in one group in *V* → goes to *a*
      * not in one group in *V* → goes to *b*
* If a pair is found not in one group in *U* and is found:
  + in one group in *V* → goes to *c*
  + not in one group in *V* → goes to *d*

Pairs sum *a* is the ground of similarity between *U* and *V*, and pairs sum *d* can be the ground of similarity in some measures; *b* and *c* attenuate similarity.

The following object pairs (or comembership) measures are available:

DICE - **Dice** similarity aka **F1** or **F** measure.

where

is called comembership **Precision**, and

is called comembership **Recall**.

*DICE* (F measure) is the harmonic mean of recall *R* and precision *P*, which, in return, can be called the two “asymmetric versions” of F measure. Under ASYMM=NO is returned, but under ASYMM≠NO the macro returns value *R* as , and value *P* as .

OCHIAI - **Ochiai** similarity aka **Folkes–Mallows** is the geometric mean of recall *R* and precision *P*.

Under ASYMM=NO is returned, but under ASYMM≠NO the macro returns value *R* as , and value *P* as .

KULCZ2 - **Kulczynski 2** similarity is the arithmetic mean of recall *R* and precision *P*.

Under ASYMM=NO is returned, but under ASYMM≠NO the macro returns value *R* as , and value *P* as .

JACCARD - **Jaccard** similarity

SS2 - **Sokal-Sneath 2** similarity

SS1 - **Sokal-Sneath 1** similarity

RAND - **Rand** aka **Simple matching** similarity

ARAND - **Adjusted Rand** similarity is *RAND* normalized in a stochastic sense under the assumption of hypergeometric distribution. The upper bound of *ARAND* for the observed pair of partitions equals 1 (if the number of groups is equal in the partitions), and the index is close to zero if both partitions of the objects into groups are independent and random (only group sizes are fixed). *ARAND* can take on negative values.

This measure is identical to **Cohen’s kappa** for 2×2 table.

RR - **Russel–Rao** similarity

RT - **Rogers–Tanimoto** similarity

PHI - **Phi correlation** similarity aka **normalized** **Hubert Г** statistic. This is the Pearson correlation in situation of binary data. It can vary in the range from -1 to 1.

DISPER - **Dispersion** similarity. Can vary in the range from -1 to 1.

SEUCLID - **squared Euclidean distance** aka **Hamming distance** dissimilarity. It is a metric distance, and is linearly equivalent to . 2*SEUCLID* is known as **Mirkin distance**.

MN - **McNemar distance** dissimilarity. This is the square root of McNemar’s test statistic.

**B)** Measures based on **frequency crosstabulation of objects**. Let there be *N* objects and two being compared partitions into groups, partition *U* (with groups 1, 2,…, *I*) and partition *V* (with groups 1, 2,…, *J*). We have the crosstabulation:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | *V* | | | |  |
| *U* |  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |  |

The following crosstabulation measures are available:

OVL - **Overlap** similarity aka **Purity**, asymmetric measure under ASYMM≠NO:

and

Its symmetric version under ASYMM=NO:

appears to be the equivalent to 1-(**Van Dongen Overlap distance**). 1-*OVL* is a metric distance.

FCA - **F Clusteting Accuracy** similarity, more sophisticated measure of overlap than *OVL*, accounting together for notions of “homogeneity” and “completeness”. (Perfectly *homogeneous* relation of *V* to *U* is when a group from *V* contains objects *only* of one group from *U*. Perfectly *complete* relation of *V* to *U* is when a group from *V* contains *all* objects of some group from *U*.) *FCA* is asymmetric, under ASYMM≠NO equal to:

and

where is cell F1 measure, equal to the harmonic mean:

,

Symmetric version of *FCA* under ASYMM=NO:

MI - **Mutual Information** similarity, an entropic measure.

is the entropy in partition *U*

is the entropy in partition *V*

is the joint entropy of *U* and *V*

Then

VI - **Variation of Information** distance is *MI* translated into the metric distance by formula:

NMI - **Normalized Mutual Information** similarity is *MI* normalized so that the possible range of the measure for the two *given* partitions *U* and *V* be from 0 to 1:

, where upper bound, , is selected (by NORMZ subcommand) equal to:

- with NORMZ=MAX and by default,

- with NORMZ=MIN,

- (arithmetic mean) with NORMZ=MEAN,

- (geometric mean) with NORMZ=GMEAN

AMI - **Adjusted Mutual Information** similarity is *MI* normalized in a stochastic sense. Its upper bound for the observed pair of partitions equals 1 (as in *NMI*), but *AMI* is close to zero in case both partitions into groups are purely random (only group sizes are fixed). *AMI* can take on negative values.

where is defined as in *NMI*, while is the expected magnitude of *MI* when *U* and *V* are two independent, random disposition of objects into groups (group sizes are set there and there).

(in this formula *x* is integer running over values “from”[[2]](#footnote-2) and “to”).

HCV - **Homogeneity&Completeness V** similarity, an entropic measure accounting for both notions “homogeneity” and “completeness” (see *FCA* for comparison), which it formulates as follows:

, and with ,

, and with ,

where the conditional entropies (see measure *MI* earlier):

and

Harmonic mean of *h* and *c*, a symmetric measure, is *HCV*:

If asymmetric measure is requested (ASYMM≠NO), then *c* and *h* are returned instead of :

and

Sources:

* Wagner, S., Wagner, D. Comparing clusterings - an overview // *A manuscript: Univ., Fak. für Informatik*, Janyary 12, 2007.
* Vinh, N.X., Epps, J., Bailey, J. Information theoretic measures for clusterings comparison: variants, properties, normalization and correction for chance // *Journal of Machine Learning Research*, 2010, 11, 2837-2854.
* Vinh, N.X., Epps, J., Bailey, J. Information theoretic measures for clusterings comparison: is a correction for chance necessary? // *Proceedings of the 26 th International Conference on Machine Learning*. Montreal, Canada, 2009.
* Rosenberg, A., Hirschberg, J. V-measure: a conditional entropy-based external cluster evaluation measure // *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*. Prague, June 2007, 410–420.
* Romano, S. et al. Adjusting for chance clustering comparison measures // *Journal of Machine Learning Research*, 2016, 17, 1–32.
* Meila, M. Comparing clusterings – an information based distance // *Journal of Multivariate Analysis*, 2007, 98, 873–895.
* Desgraupes, B. Clustering indices // *Package clusterCrit for R*, April 2013.
* Rendon, E. et al. Internal versus external cluster validation indexes // *International Journal of Computers and Communications*, 2011, 1 (1), 27–34.
* Halkidi, M., Batistakis, Y., Vazirgiannis, M. On clustering validation techniques // *Journal of Intelligent Information Systems*, 2001, 17 (2/3), 107–145.

**NORMZ**

This subcommand is needed to compute NMI and AMI (see s/c MEASURE), it sets the method of normalization. Select one of: MAX (also default), MIN, MEAN, GMEAN.

**PRINT**

Specify PRINT=YES if you want to see some technical details, namely, values A, B, C, D from comembership confusion matrix (type “A” measures are computed from them) and frequency crosstabulations (type “B” measures are computed from them). In the frequency crosstabs, variable values (group codes) are not displayed by the macro, and their order is the order the values were encountered in the data, not the sorted order of them. By default, PRINT=NO.

***Special regimes***

The macro ignores weighting of the dataset (however, it doesn’t take in the procedure cases with missing and nonpositive weights) and is not suited for splitting the dataset (SPLIT FILE). The macro obeys case selection/filtering (commands FILTER, USE, SELECT IF, N OF CASES) and transforms, including those under TEMPORARY command.

# MACRO !KO\_CLASAGREE: COMPARISON MEASURES FOR “CLASSIFICATION” PARTITIONS

Version 1, Mar 2019. Tested on SPSS Statistics 17, 20, 22.

!KO\_clasagree vars= v1 WITH v2 to v6 /\*Variables representing case groupings;

/\*word WITH is possible after the first name

/measure= ACC F1 OSR /\*Specify measure(s): ACC, REC, PRE, SPE, NPV, YOUD, MARK,

/\*F1, FBETA, KULCZ2,

/\*LNDOR, DP, CORR, BCR, GM, AGM, OPRE, JACCARD, OSR, KAPPA, SPI, BPK, RKCORR

/binar= /\*Treatment for binary measures: CLASS, MEAN (default), WMEAN, IWMEAN

/beta= 0.5 /\*Beta for FBETA measure

/print= YES /\*Show technical details: NO (default) or YES.

Minimal specification VARS, MEASURE.

The macro computes various measures of agreement between partitions with “labeled” (identified) groups. These measures are known as *classification agreement indices*, measures of *performance quality of classifiers*, and other names. In the being compared partitions *U* and *V* **there are knowingly mutually correspondent (“same”) groups, by name**. This reflexes on the data coding, which must be uniform. A group with code, say, 1 in *U* is the “same” group as the group with code 1 in *V*; group with code 2 in *U* is the “same” group as the group with code 2 in *V*, and so on. Consequently, *U* and *V* agree well (are similar) if there prevail objects falling in them in groups with the same code: 1 in *U* and 1 in *V*, 2 in *U* and 2 in *V*. Word “partition” is a synonym of g-class grouping, classes not intersecting.

Whereas if mutually correspondent groups are not identified in the partitions, that is, it is unknown to which group in *U* the group coded as 1 (for example) in *V* corresponds, then use macro [!KO\_CLUAGREE](#_МАКРОС_!PROXQNT:_РАЗНЫЕ). Or establish first the corresponding groups by some way, for example empirically by macro [!KO\_GRMATCH](#_МАКРОС_!GRMATCH:_ИДЕНТИФИКАЦИЯ_1).

For macro !KO\_CLASAGREE, groups can be any by origin: they could be results of classifications, clusterings, natural or exemplar groupings. Code match between partitions can be not complete and the number of groups in partitions can be not equal. Say, *U* can consist of groups 1, 2 and 3, and *V* – of groups 1, 2, 4 and 5. In this instance correspondent groups are two (pairs) – 1 and 2, they exist in both partitions; other groups are noncorrespondent and they cannot be the ground of agreement between *U* and *V*, on the contrary – they vote for nonagreement. As usual, however, comparing classifications often utilizes data where *U* and *V* are entirely made of mutually correspondent groups.

Objects are dataset cases and partitions are variables. There can be two or more partitions, the macro compares them in twos. Partitions must be mutually exclusive groups ( = alternative groups, crisp groups), i.e. each object belonging to precisely one group (a group may consist even of one object). This macro does not produce measures for fuzzy groupings (for example, multi-topic classifications). The macro makes no use of any data features based on which the classifications/clusterings were obtained. Only object groupings is the information input to the analysis.

The macro outputs the computed measures in Viewer and in new dataset. See detail in s/c VARS.

EXAMPLE 1.

!KO\_clasagree vars= true WITH clas1 clas2 clas3 /measure= ACC F1 OSR.

* Three different partitions left after different classifiers, CLAS1, CLAS2, CLAS3, are compared each with the standard classification, TRUE, by indices: Accuracy, F1-measure, Relative Agreement (OSR).

EXAMPLE 2.

!KO\_clasagree vars= clas1 to clas10 /measure= KAPPA.

* Results of different classifications (different experts classifying objects in groups), variables CLAS1 to CLAS10, are compared each with each pairwisely by Cohen’s kappa, and the square matrix is returned.

***Subcommands***

**VARS**

Specify input numeric categorical, nominal variables (names up to 8 bytes long) which are the being compared partitions of the objects (the dataset cases). Each unique value in a variable signifies a group. At least two variables; you can use word “to” in the list for contiguous variables in the dataset. The variables must have no missings (if missings, cope with them in advance: you could decide to exclude objects with missings or enlist them in a single new group or each such object in its own group). Each variable must be not a constant and must have at least one duplicating value (that is, the number of groups must be greater than one and less than the number of objects). The macro does not check the correctness of the input variables, so you should watch yourself.

There are two modes to specify VARS: with keyword WITH and without it. If without WITH, for instance:

VARS= *CLAS1 CLAS3* to *CLAS10*

Then you have the right to request only one proximity measure in s/c MEASURE. The macro will compute that proximity measure between all partitions VARS pairwisely and will return, as a new dataset, the square matrix with the proximity values. If the measure is symmetric, there will be complete matrix at the output. If the measure is asymmetric, there will be matrix with filled above-diagonal triangle at the output. Cell (i,j) in a matrix is the result of comparison of ith variable as partition *U* with jth variable as partition *V*. See EXAMPLE 4 if you need to fill in the below-diagonal triangle as well. There stands 1 on the diagonal if the upper bound for the proximity is 1, and there stands arbitrary number 999 if the upper bound for the proximity is +∞.

If to use word WITH, then it must go *second*, after the name of one partition that we denote *U*. Every partition following the word WITH, we denote *V*. So, one specified *U* will be compared with each of *V*. If you are comparing one reference (standard) classification with one or several predicted classifications, the reference one should be *U*. For example:

VARS= *TRUECLAS* WITH *CLAS1 CLAS3* to *CLAS10*

Partition *TRUECLAS* will be compared with each listed after WITH. This is the comparison of one partition with all the rest. You can request more than one proximity measure in s/c MEASURE. The macro will return the rectangular result matrix in a new dataset and in Viewer window.

**MEASURE**

Select a proximity measure or measures (in the presence of WITH in VARS list) from the list. The macro computes two types of measures: measures for **binary** classification (these measures’ values the macro can average into values of multiclass classification) and measures for **multiclass** classification **proper**.

Both these and those measures proceed from the frequency crosstabulation of the being compared partitions (classifications) *U* (*I* groups) и *V* (*J* groups):

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | *V* | | | |  |
| *U* |  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |  |

Fig. 1.0

Names (labels) of classes are known, and they are partly or completely same in *U* and *V*. (If *U* and *V* do not share even a single class, the macro will not compute measures for this pair of partitions and will notify of that.) Here are examples of crosstabulations:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  | *V* | | | |
|  |  | *class 1* | *class4* | *class 2* | *class 3* |
| *U* | *class 1* | 12 | 0 | 6 | 3 |
| *class 3* | 7 | 10 | 5 | 8 |
| *class 2* | 2 | 1 | 21 | 0 |

Fig. 1.1

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | *V* | |
|  |  | *class 1* | *class 2* |
| *U* | *class 3* | 1 | 2 |
| *class 2* | 9 | 28 |
| *class 4* | 3 | 10 |
| *class 1* | 35 | 0 |

Fig. 1.2

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | *V* | | |
|  |  | *class 1* | *class 2* | *class 3* |
| *U* | *class 1* | 42 | 11 | 7 |
| *class 2* | 0 | 14 | 1 |
| *class 3* | 18 | 5 | 26 |

Fig. 1.3

Cells on the crossing of classes of the same name (i.e. groups coded in variables *U* and *V* by some same code) in the above examples are flagged (shaded). Note that in general case it is not necessary that *U* and *V* consist completely of the same classes, as it is on Fig. 1.3, where both partitions are exhausted by classes 1, 2, 3. On Fig. 1.1, *U* consists of three classes, but *V* consists of four, three classes (conventionally designated 1, 2, 3 here) – are common for both. Analogously, on Fig. 1.2 *U* consists of four classes, but *V* – of two, these two classes being common in *U* and *V*. Sorting of rows and columns in the cross-table by class codes does not play a role in the analysis. So, on Fig. 1.1 and 1.2 class order isn’t sorted ascendingly. Marginal row and column counts are always nonzero since the macro produces tables only based on the present, observed data. Minimal table size is 2x2.

We’ll designate by letter *M* the set of classes common in *U* and *V*, and by letter *K* the number of these common classes, 1≤*K*≤min(*I,J*). While *N* is the sum of frequencies of the whole table (= the number of the dataset cases), not only in the *K* cells flagged in it.

Measures of agreement (similarity) between *U* and *V* can be symmetric or asymmetric. A *symmetric* measure will not change value if *U* (table rows) and *V* (table columns) swap their places, i.e. the frequency cross-table get transposed. An *asymmetric* measure will change value after the table is transposed[[3]](#footnote-3). **For asymmetric measures it is significant** which of the two partitions to consider *reference* (exemplar, true), and which to consider *predicted* (experimental). Macro !KO\_CLASAGREE **treats partition *U* (defining table rows) as the reference classification, and partition *V* (defining table columns) as the predicted classification**.

**A**) Measures for **binary** aka **one-hot** aka **class-specific** classification. The focus of interest is a classification in one specific class.

Such agreement (similarity) measure is calculated for each class *k* of *M*. The macro builds for the class the **2x2 confusion matrix**, where partition *U* (rows) is taken for reference classification, and partition *V* (columns) is taken for predicted classification:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | *V* | |  |
|  |  | *class k* | *not class k* |  |
| *U* | *class k* | True Positives  *a* | False Negatives  *b* | *a+b* |
| *not class k* | False Positives  *c* | True Negatives  *d* | *c+d* |
|  |  | *a+c* | *b+d* | *a+d+b+c = N* |

This matrix is the *I*x*J* frequency crosstabutation dichotomized. Based on counts *a*, *b*, *c*, *d* this or that measure is computed. Because there are *K* such confusion matrices, there are *K* values of the measure. Below are the formulas of different measures as computed for each class. Subcommand BINAR=CLASS returns all *K* values, one for each class from *M*. With other specification of BINAR subcommand (see there) the *K* values are averaged in one value that characterizes the agreement between *U* and *V* overall.

Some binary measures are symmetric, other are asymmetric – under BINAR=CLASS or MEAN (symmetric measures don’t change value if swap *U* and *V*). Under BINAR=WMEAN or IWMEAN all binary measures become asymmetric because weighting is done always by class sizes of the partition that forms *rows* of crosstabulation and of confusion matrix. The following binary measures are available (they all are similarities):

ACC - **Accuracy** aka **Rand** aka **Simple Matching** coefficient, symmetric, ranges [0,1].

1-*ACC* is called **Missclassification** (or **Error**) **Rate** and is linearly equivalent to the squared euclidean distance.

REC - **Recall** aka **Sensitivity** aka **True Positive Rate** aka **Hit Rate** aka **Positive Accuracy**, asymmetric (counterpart value =*PRE*), ranges [0,1].

SPE - **Specificity** aka **True Negative Rate** aka **Negative Accuracy**, asymmetric (counterpart value =*NPV*), ranges [0,1].

YOUD - **Youden's index**, asymmetric, ranges [-1,1].

PRE - **Precision** aka **Positive Predictive Value**, asymmetric (counterpart value =*REC*), ranges [0,1].

1-*PRE* is called **False Discovery Rate.**

NPV - **Negative Predictive Value**, asymmetric (counterpart value =*SPE*), ranges [0,1].

MARK - **Markedness index**, asymmetric, ranges [-1,1].

F1 - **F1** aka **F Measure** aka **Dice Matching** coefficient, symmetric, ranges [0,1], it is the harmonic mean of *REC* and *PRE*.

With zero *REC* and *PRE* the measure is not computed.

FBETA - **F-beta** aka **generalized** or **weighted F Measure**, generally asymmetric, ranges [0,1], it is the weighted harmonic mean of *REC* and *PRE*.

,

where parameter *beta* (0,+∞) is set by BETA subcommand. If *beta*<1, *PRE* receives greater weight than *REC*; if *beta*>1, then vice versa, *REC* receives greater weight. At *beta*=1 *FBETA* turns into *F1*. With zero *REC* and *PRE* the measure is not computed.

KULCZ2 - **Kulczynski 2** coefficient, symmetric, ranges [0,1], it is the arithmetic mean of *REC* and *PRE*.

2*KULCZ2*-1 is called **Classification Success Index**.

LNDOR - **logarithm of Diagnostic Odds Ratio**, symmetric, ranges [-∞,+∞].

If *a*, *b*, *c* or *d* is zero, the measure is not computed.

DP - **Discriminant Power**, symmetric, ranges [-∞,+∞].

If *REC* or *SPE* is 1 or 0, the measure is not computed.

CORR - **Matthews correlation** aka **Phi correlation**, symmetric, ranges [-1,1]. This is nothing else than Pearson correlation in case of binary data.

,

where and .

BCR - **Balanced Classification Rate**, asymmetric, ranges [0,1], it is the arithmetic mean of *REC* and *SPE* and it is the AUC (area under the curve) in the ROC space, for a single point there.

GM - **GM Measure**, asymmetric, ranges [0,1], it is the geometric mean of *REC* and *SPE*.

AGM - **Adjusted GM Measure**, asymmetric, ranges [0,1], is a modification of *GM* designed to better cope with unbalanced classes.

, and with *REC*=0, *AGM*=0

where

If the positive (binary) class is disproportionally small, this measure may be preferable to *GM* because it is more sensitive to changes in *SPE* than to changes in *REC*.

OPRE - **Optimized Precision**, asymmetric, ranges [-∞,1]. The measure is higher when *a* and *d* both are high.

If *REC*+*SPE*=0, the measure is not computed.

JACCARD - **Jaccard** aka **Tanimoto** Matching coefficient, symmetric, ranges [0,1].

Measures ACC, PRE, NPV, CORR, F1, FBETA, OPRE, JACCARD are sensitive to change in the shape of distribution of objects in classes of reference (*U*) classification, that is why it is said that these measures are not the same result in situations of balanced and unbalanced classes.

**B**) **Multiclass proper** aka **nominal** classification measures. These are symmetric measures of association for two nominal variables, and are known also as measures of experts agreement.

Such a measure, its single value, is computed immediately from the complete frequency crosstabulation (see Fig. 1.0–1.3). The basis of similarity is the frequencies in the cells on the crossings of the same-named classes, i.e. the *K* cells constituting the set *M*. The following multiclass measures are available, they all are similarities:

OSR - **Overall Success Rate** aka **Relative Agreement** aka **Multiclass Accuracy** is simply the proportion of correctly (samely) classified objects, the proportion of objects that fall in *U* and *V* in groups of the same code.

KAPPA - **Cohen’s Kappa** is *OSR* normalized by level of random agreement. Ranges in [-∞, 1]. Value 0 means agreement on a random level.

, where observed agreement , and expected random agreement is

SPI - **Scott’s Pi** is another way to normalize *OSR* by level of random agreement. Ranges in [-∞, 1]. Value 0 means agreement on a random level. Formula of *SPI* is the same as of *KAPPA*, but

BPK - **Brennan–Prediger Kappa** – one more way to normalize *OSR* by level of random agreement, without assuming fixed class sizes (i.e. of marginal frequencies). Ranges in [-∞, 1]. Value 0 means agreement on a random level. Formula of *BPK* is the same as of *KAPPA*, but

If *I=J=K*, then and *BPK* is linearly equivalent to *OSR*.

RKCORR - **Multiclass Matthews correlation** aka **Rk correlation**. It is Pearson correlation computed bijectively between two sets of dummy variables, ranges in [-1, 1]. If to turn two categorical variables consisting of the same *K* categories into two sets of binary dummy variables, *K* columns each, where the columns – they represent the categories – are co-ordered between the sets, then after centering all the columns one should vectorize each *N*x*K* set into the column of length *NK*. Then Pearson r computed by the “cosine formula” between the two obtained columns X and Y is *RKCORR*:

, where is the sum of cross-products of X and Y, and and are the sums of squares in the two columns.

Equivalent formula to compute *RKCORR* from the square *K*x*K* crosstabulation with the same, co-ordered categories (see Fig. 1.3 above):

*RKCORR* is not computed if partitions *U* and *V* consist of not completely the same sets of group codes.

Sources:

* Tharwat, A. Classification assessment methods // *Applied Computing and Informatics* (2018), https://doi.org/10.1016/j.aci.2018.08.003.
* Labatut, V., Cherifi, H. Evaluation of performance measures for classifiers comparison // *Ubiquitous Computing and Communication Journal,* 2011, 6, 21-34.
* Hossin, M., Sulaiman, M. A review on evaluation metrics for data classification evaluations // *International Journal of Data Mining & Knowledge Management Process (IJDKP),* 2015, 5(2), 1-11.
* Sokolova, M., Lapalme, G. A systematic analysis of performance measures for classification tasks // *Information Processing and Management*, 2009, 45, 427–437.
* Matthews, B.W. Comparison of the predicted and observed secondary structure of T4 phage lysozyme // *Biochimica et Biophysica Acta (BBA) - Protein Structure*, 1975, 405(2), 442–451.
* Gorodkin, J. Comparing two K-category assignments by a K-category correlation coefficient // *Computational Biology and Chemistry*, 2004, 28, 367-374.
* Jurman, G., Riccadonna, S,, Furlanello, C. A comparison of MCC and CEN error measures in multi-class prediction // *PLoS ONE*, 2012, 7(8): e41882, doi:10.1371/journal.pone.0041882.
* Brennan, R., Prediger, D. Coefficient Kappa: some uses, misuses and alternatives // *Educational and Psychological Measurement*, 1981, 41, 687-699.
* Maxwell, A.E. Coefficients of agreement between observers and their interpretation // *Brit. J. Psychiatry*, 1977, 530, 79-83.
* Batuwita, R., Palade, V. Adjusted geometric-mean: a novel performance measure for imbalanced bioinformatics datasets learning // *Journal of Bioinformatics and Computational Biology*, 2012, 10(4), 1250003-1­ - 1250003-23.

**BINAR**

This subcommand is for binary comparison measures (see s/c MEASURE, measures “**A**”), it is ignored by multiclass measures (measures “**B**”). At a two partitions comparison, binary measures are computed first for each of the *K* shared classes. BINAR=CLASS outputs these class-wise values in the result. BINAR=CLASS is incompatible with specification of any of multiclass measures. BINAR=CLASS is allowed only with keyword WITH used in VARS.

Other options of BINAR imply averaging *intraprocess* or *after* computing a measure (see below), and the macro thus will output a single value characterizing the degree of agreement between the two partitions. Averaging methods:

MEAN - (also default/unspecification) simpe arithmetic mean of the *K* values.

WMEAN - weighted arithmetic mean of the *K* values. Weights are the *row* marginal frequencies, i.e. the class sizes in partition *U*. Thus, classes of greater sizes bear more on the averaged result. Use it if the quality of prediction for large classes is more important to you.

IWMEAN - weighted arithmetic mean of the *K* values. Like the former, only weights are the inverted row marginal frequencies, therefore classes of smaller sizes bear more on the averaged result. Use it if the quality of prediction for small classes is more important to you.

With BINAR= WMEAN or IWMEAN all binary comparison measures become asymmetric, because weights are always the sizes in the *reference* classification (the one that defines *rows* of crosstabulation).

*When the averaging takes place*. For the following measures, *K* values get averaged *after* the measure is computed: ACC, REC, PRE, SPE, NPV, LNDOR, DP, JACCARD. The *K* values of the measure get averaged.

For the following measures, averaging is done *before* the measure is computed with its formula: YOUD, MARK, F1, FBETA, KULCZ2, CORR, BCR, GM, AGM, OPRE. For these measures, there are averaged *K* values of each *term* of the formula, and only then one value of the measure is computed by its formula. You can see the terms on level of which averaging is performed in the formulas (see s/c MEASURE). For the majority of such measures, formula terms are: *REC*, *PRE*, *SPE*, *NPV*. For example, to obtain an averaged F1 measure, first *K* values of *PRE* get averaged, likewise *K* values of *REC* get averaged. The emergent average *PREM* and *RECM* are substituted into the formula of F1. For measure CORR, the formula terms which each undergo averaging are *a/N*, *S* and *P*; for measure AGM they are *REC*, *SPE* and *Q*.

Note that the averaged binary measure CORR is not the same thing as multiclass measure RKCORR, albeit both this and that are based on Pearson correlation.

Measure DP has *REC* and *SPE* as its terms, still it is averaged after the formula and not before the formula, so is done in order to support the measure’s symmetry.

EXAMPLE 3.

!KO\_clasagree vars= class WITH clas1 clas2 /measure= REC PRE SPE NPV /binar=CLASS /print=YES.

* Experimental classifications CLAS1 and CLAS2 are assessed relative the ideal classification CLASS. Requested, for each existing class, basic asymmetric measures of such assessment. Also, demanded to display (PRINT=YES) are frequencies based on which those measures were computed.

**BETA**

This subcommand is used only by measure FBETA(see the formula). Specify, as parameter *beta*, number greater than zero. If *beta*<1, *PRE* receives greater weight than *REC*; if *beta*>1, than vice versa, *REC* receives greater weight. At *beta*=1 FBETA measure turns into F1 measure. Usually values 2 or 0.5 are specified.

**PRINT**

Specify PRINT=YES if you want to see some technical details, namely, frequency crosstabulations. In the frequency crosstabs, the class order is not sorted, it is the order the values were encountered in the data. If binary comparison measures were requested, the macro shows also all the class-wise counts TP(*a*), FN(*b*), FP(*c*), TN(*d*). By default, PRINT=NO.

EXAMPLE 4. Obtaining both matrix triangles for an asymmetric measure. Dataset Data is being analyzed.

!KO\_clasagree vars= p1 p2 p3 p4 p5 /measure= BCR.

dataset name mx1.

* BCR is an asymmetric measure, therefore in the output only above-diagonal triangle is filled in the matrix. In cell (1,2), for instance, there contains value BCRP1P2, where P1 were U partition (defining rows of crosstabulation), and P2 were V partition (defining columns of crosstabulation). We wish to obtain the counterpart values, such as BCRP2P1, which would fill in the below-diagonal triangle of the matrix. The trick is this:

compute casenum= $casenum.

dataset activate Data.

!KO\_clasagree vars= p5 p4 p3 p2 p1 /measure= BCR.

compute casenum= $casenum.

sort cases by casenum (D).

compute casenum= $casenum.

execute.

update file= mx1 /file= \* /by casenum.

execute.

delete var casenum.

dataset close mx1.

* First, created case indicator variable CASENUM in the returned matrix. Then turned back to Data and ran the macro once again, now with list VARS written in the reverse order. The macro output the seeked for values, however they are found again in the upper triangle. Therefore here indicator CASENUM was created too, and the cases were sorted in the reverse order by it. Having done it, recalculated CASENUM (thus the last case, which appeared the first one after the sorting, has now CASENUM=1). After that, UPDATE command combined data of this matrix with that of the first one (MX1). In the result, we have both triangles of the matrix correctly filled.

***Special regimes***

The macro ignores weighting of the dataset (however, it doesn’t take in the procedure cases with missing and nonpositive weights) and is not suited for splitting the dataset (SPLIT FILE). The macro obeys case selection/filtering (commands FILTER, USE, SELECT IF, N OF CASES) and transforms, including those under TEMPORARY command.

# MACRO !KO\_GRMATCH: IDENTIFICATION (PAIRING) OF MATCHING GROUPS

Version 1, Mar 2019. Tested on SPSS Statistics 17, 20, 22.

!KO\_grmatch vars= v1 WITH v2 to v6 /\*Variables representing case groupings;

/\*word WITH is possible after the first name

/matchon= /\*Match on the basis of counts (COUNT, default), residuals (RESID),

/\*or F measures (FCELL)

/method= GREEDY /\*Matching algorithm: Hungarian (HUNGAR), greedy (GREEDY),

/\*or greedy with prohibition (PGREEDY)

/recode= 'd:\exercise\RecSyntax.sps' /\*Whether to recode the variables: NO (default),

/\*YES, or path/name of sps-file

/other= /\*Non-matched groups: leave individual (INDIVID, default) or merge

/\*(specify group code)

/simmx= /\*Create matrix of matches of groups and show on the dendrogram: YES

/\*or NO (default)

/print= /\*Report about matching in Output: YES (default) or NO.

Minimal specification VARS, METHOD.

Let you have a dataset of cases and two or more partitions of this dataset into groups of cases (objects). The partitions could be the results of clustering, classification, other grouping by origin (including natural or true/ideal grouping). You expect that the partitions are same or, at least, are quite similar – in the sense that cases found in one group in one partition are found predominantly in one group also in another partition. The problem with the data is however that the codes, or “group labels”, are not the same in the partitions. For instance, partitions *U* and *V* are same or similar and consist of groups A, B and C, but these groups are *coded* in *U* with values 1, 2, 3, and in *V*, respectively, with values 2, 6, 4. You *don’t know which* group in *V* corresponds to *which* group in *U*, corresponds **in the sense that it consists mainly of the same cases of the dataset**. Identification of this, empirical correspondence – of mutual match between the groups of different partitions – is the task of the macro. Word “partition” is the synonym of g-class grouping (nonoverlapping classes).

Partitions are categorical, nominal variables. Each case belongs to precisely one group (class, cluster etc.) in every partition, i.e. a partition is one variable in the dataset and the groups of it are alternative. A group may consist of a single object. The macro’s task is to compare the partitions pairwisely for the issue which groups in them do the same cases fall into. If primarily the same cases fall in group A of partition *U* as those that fall in group X of partition *V*, – then A and X are approximately “one the same” group, or, in other words, A and X are matching groups. Let *U* contain 3 groups but *V* contain 5 groups. The macro will pair three groups in *U* with three groups in *V*; this is *one-to-one* pairing: one group with only one group. Two “excess” groups of *V*, that the macro will judge insufficiently similar to the groups of *U*, will be left unpaired.

Variables (partitions) are juxtaposed in twos. Group pairing (identification of the matching) between them is done based on their crosstabulation (frequencies, residuals or F measures) by a pairing algorithm trying to maximize the sum in the matched pairs.

The macro can do, or can write out as syntax file, recoding of the variables, so that same/similar grops in them be coded identically (have the same “group label”). You can also see groups recognized as correspondent (matched) by the macro on a dendrogram.

***Subcommands***

**VARS**

Specify the list of categorical numeric variables (names up to 8 bytes long), you may use “to” in it. Missings are not allowed in the data. Each variable is a partition of cases of the dataset into groups. Different variables are such different partitions. The macro will compare variables by two and establish which groups of one partition are correspondent to which groups of the other partition. Each unique value in a variable signifies a group: the variables are treated as nominal. It doesn’t matter if values (group codes) used in different variables are same or different.

*Permissible coding*. The values may have any sign, be integer or fractional. Value width (including negative sign and decimal separator, if present) – up 8 digits. If the code is fractional it must have *no more than two* significant decimal digits. The macro does not check input values, so you watch yourself that the values do not violate the described requirements.

*Keyword WITH*. After *the first* name in VARS you may put, once, keyword WITH. Then each of the remaining variables VARS will be compared with one variable – that, which is mentioned before WITH. If there is no WITH word in the list, then all the variables will be compared with each other pairwisely.

EXAMPLE 1.

!KO\_grmatch vars= true WITH clu1 clu2 clu3 /method= GREEDY /recode= 'd:\exercise\RecSyntax.sps'.

* Three different cluster solutions (obtained by different clustering methods), CLU1, CLU2, CLU3, are compared each with standard classification TRUE. Greedy pairing method is used to establish mutually corresponding groups. Requested is to write out syntax, to recode variables CLU1, CLU2, CLU3, in order to make their group codes uniform with that in variable TRUE.

EXAMPLE 2.

!KO\_grmatch vars= p1 to p8 /method= HUNGAR /matchon= FCELL /simmx= YES /print= NO.

* All alternative groupings from P1 to P8 are compared with each other pairwisely in search of the equivalent groups between them. The search is done by Hungarian matching algorithm on the basis of F measures. Results – “same” groups from different groupings – are requested to show graphically on a dendrogram. Output of detailed results is suppressed.

**MATCHON**

Select which table to display in the output as “Crosstabulation”, on the basis of whose entries, values in cells *ij*, the pairing between its rows (groups of one variable) and its columns (groups of the other variable) will be done:

COUNT - (default/unspecification) table of counts: . That means that for the evidence of one-to-one match you are taking a high, comparatively to table-average, observed frequency in a cell.

RESID - table of frequency residuals: . That means that for the evidence of one-to-one match you are taking cell frequency that is high comparatively to the expected frequency in that cell. Expected frequency in the cell is the frequency completely determined by the pair of its marginal frequencies and , i.e. the volumes of the two being considered groups. When the observed intersection of the two groups is big but it is big only because these groups are themselves big, then this doesn’t become too weighty argument for their matching.

FCELL - table of cell F measures: . This is one more way to operate not by frequencies as they are but by their relations with marginal frequencies. F measure in a cell is the harmonic mean of the cell’s “recall” , and “precision” .

**METHOD**

Select matching algorithm.

HUNGAR - Hungarian (Kuhn–Munkres) algorithm. It is iterational and it maximizes the sum of values of the selected cells (i.e. in paired rows and columns) to the global optimum.

GREEDY - simple greedy algorithm. It is stepwise and is faster, but does not guarantee to reach the global optimum, the maximal possible sum of values in the selected cells, albeit it aims to maximize it. On each step, the algorithm selects the largest currently element in the table, considers it to be the pairing done, and removes that row and that column from the table. This approach picks the most big values from the first steps, while Hungarian is more “prudent” and may sometimes prefer not the biggest element if it will yield the bigger sum in the end.

PGREEDY - greedy algorithm with prohibition. This is the same as GREEDY but imposes an additional condition: the largest currently element in the table considers a done pairing only if it is maximal in its row and in its column in the initial table. Otherwise the pairing on this step doesn’t reckon, althogh the row and the column become deleted. This variant thus permits matching only due to most big values in rows and columns of the crosstabulation. It is nothing more than GREEDY in which “unconvincing” pairings are simply abolished.

If in two being compared partitions the number of groups is *I* and *J*, then HUNGAR and GREEDY always form *min(I,J)* pairs, but PGREEDY can give less pairs.

**RECODE**

This subcommand enables to recode variable VARS following the word WITH into the correspondence to the variable standing prior WITH. The recoding is dictated by the correspondence found during the matching. Each group with some code *a* in variable X (going after WITH) changes its code to code *b* if in variable I (that is before WITH) that group has the one-to-one corresponding group coded *b*. In other words, the recoding directly follows table “Matchings” displayed in the output. Choose:

NO - (default/unspecification) don’t undertake recoding.

YES - do recoding (of the VARS variables standing after WITH).

*filename* - don’t do recoding, but write its syntax. Specify (in quotes or apostrophes) path/name of SPS-file for saving. Using this syntax you can do the recoding later or recode another dataset with it. (If the recoding actually was not needed, the syntax file will contain recoding of values into themselves.)

RECODE= YES or *filename* is permitted only having the keyword WITH in VARS.

**OTHER**

This subcommand plays role with RECODE= YES or ‘file’. It defines how to do, at the recoding, with those groups of variable X which miss matching groups to pair with in variable I.

INDIVID - (default/unspecification) keep these groups individual.

*value* - merge these groups in one group. Specify a code for the group. See requirements for permitted codes in VARS subcommand.

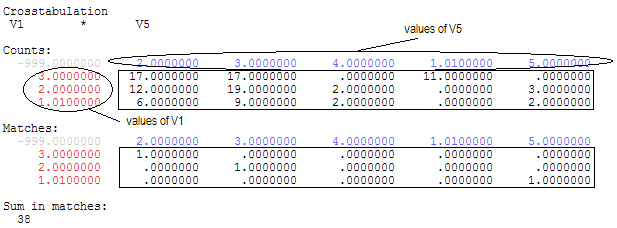
When OTHER=INDIVID, the macro preserves individuality of nonpaired groups the following way. If no other group pretends for the code of such group at the recoding, the group retains its code. But if some other group pretends for its code, then the code is modified by adding 0.01 to it repeatedly until the code becomes again unique, i.e. there are no duplicate claimants for it.

With METHOD=PGREEDY, pairings that were not approved because they did not satisfy the restriction-condition (see), are considered, too, as groups of variable X for which there are no matched groups in variable I, so s/c OTHER treats them similarly.

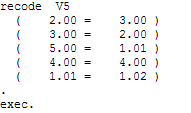
EXAMPLE 3.

!KO\_grmatch vars= v1 WITH v5 /method= HUNGAR /recode= 'd:\exercise\RecSyntax.sps'.

* Partition V1 is compared with partition V5. Because WITH is present, it is possible to write out syntax for recoding of groups V5 into coding of groups present in V1. In Viewer output the macro returned:



* Cross-table of counts, in which values of the two variables –V1 codes are the rows, and V5 codes are the columns. The macro does not sort values ascendingly, so they are shown in the order they were met in the data. We see that in V1 there are three groups with codes 1.01, 2, 3, and in V5 there are five groups with codes 1.01, 2, 3, 4, 5.
* The macro’s task is to pair groups between the two partitions based on this table of frequencies. The grounds for a pairing is a high count in a cell of the table (pairing tries to maximize sum within pairs, Sum in Matches). Results of the pairing are shown in table Matches. Group 2 (V5) is paired (i.e. recognized as mutually correspondent) with group 3 (V1), group 3 (V5) is paired with group 2 (V1), group 5 (V5) is paired with group 1.01 (V1). Two groups in V5 (4 and 1.01) were not paired with any groups of V1 (a group can be paired no more than with one group of the other variable: one-to-one match). The macro wrote out syntax for recoding of V5, following the results of matching:



* Codes 2, 3, 5 in V5 must be replaced to their corresponding codes of V1. Codes 4 and 1.01 can be left as they are, because there is no matched groups in V1 for them. However, code 1.01 is preoccupied: this code exists in V1 and code 5 of V5 will be recoded in it. Therefore the macro added 0.01 to code 1.01 of V5 in order to keep its individuality (by default, OTHER=INDIVID), so 1.01 is suggested to replace with 1.02. If happened 1.02 were also occupied the macro would have suggested 1.03, etc.

**SIMMX**

Because in the absence of keyword WITH in VARS there are compared all variables VARS each with each pairwisely, results could be gathered together in a square symmetric matrix which rows and columns are all groups from all the variables. In this matrix, matched pairs are flagged with 1 while other pairs are 0. Besides, for convenience, groups of the same partition (variable) have minus 1 on their intersection, which highlights the fact of their alternativeness, these groups by definition cannot be correspondent. Thus, the matrix is a similarity matrix with three distinct values in it.

Specify SIMMX=YES if you want to obtain this matrix of correspondence. The macro will output it as a new unnamed dataset and will do hierarchical cluster analysis by complete linkage (farthest neighbour) method. You will see the dendrogram on which mutually correspondent groups from different variables are noticeable at once. SIMMX=YES is allowed only if word WITH is absent in VARS. By default and SIMMX=NO the matrix is not built and the dendrogram is not displayed.

**PRINT**

By default and with PRINT=YES the macro reports on matched groups in pairs of variables. If you want to suppress the output of these results, command PRINT=NO; you must then specify RECODE or SIMMX.

***Special regimes***

The macro ignores weighting of the dataset (however, it doesn’t take in the procedure cases with missing and nonpositive weights) and is not suited for splitting the dataset (SPLIT FILE). The macro obeys case selection/filtering (commands FILTER, USE, SELECT IF, N OF CASES) and transforms, including those under TEMPORARY command.

1. **A decision schema** around the issue of asymmetric agreement indices in comparison of clustering partitions (i.e., no knowledge of labels is used). Indices can be symmetric or asymmetric mathematically and partitions can be symmetric or asymmetric positionally. Positionally symmetric partitions is when we're just comparing two alternative partitions. Positionally asymmetric partitions is when one of the two is "reference" and the other is "experimental" - different roles.

   If the index is symmetric and the roles are symmetric, then no problem. If the index is symmetric and the roles are asymmetric, we say: the given symmetric index serves the asymmetric roles (like, say, correlation coefficient is still a valid index in a cause-response relationship).

   If the index is asymmetric and the roles are symmetric, then we treat the index asymmetry as a mere inconvenience and average/combine the two values (if the action is defensible).

   If the index is asymmetric and the roles are asymmetric, then we may accept and interpret both distinct values of the index (plus possibly average/combine them into one. [↑](#footnote-ref-1)
2. In the source article, the lower index is given as rather than , but the effect is the same. [↑](#footnote-ref-2)
3. **A decision schema** around the issue of asymmetric agreement indices in comparison of classification partitions (i.e., knowledge of labels is used). Indices can be symmetric or asymmetric mathematically and partitions can be symmetric or asymmetric positionally. Positionally symmetric partitions is when we're just comparing two alternative partitions. Positionally asymmetric partitions is when one of the two is "reference" and the other is "experimental" - different roles.

   If the index is symmetric and the roles are symmetric, then no problem. If the index is symmetric and the roles are asymmetric, we say: the given symmetric index serves the asymmetric roles (like, say, correlation coefficient is still a valid index in a cause-response relationship).

   If the index is asymmetric and the roles are symmetric, then we refuse the symmetry of roles and move to the situation where one partition is declared "reference" and the other "experimental".

   If the index is asymmetric and the roles are asymmetric, then we interpret only one of the two values of the index, based on "who is who" in the partitions pair. [↑](#footnote-ref-3)