***Multiple Response tools***

SPSS macros by Kirill Orlov

kior@akado.ru, ttnphns@gmail.com

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

All rights reserved

*Tools for multiple response sets.* Collection of diverse macros to work with variables of multiple response sets – categorical sets (MRC) and dichotomous sets (MRD), except for recoding them one into another (look for that in “Categorical – Binary recodings” collection).

* For maintenance of a categorical multiple response set (letting it regular build, randomizing entry order, removing response duplicates, curing incompatible response conflicts, cropping away empty variables, merging sets) use [!KO\_REGMRC](#_MACRO_!KO_REGMRC:_REGULATING).
* For creation or updating of “no answer to the question” variable in a dichotomous multiple response set use [!KO\_MRDNA](#_MACRO_!KO_MRDNA:_SUPPLYING).
* If you have to clean data (add, remove responses) of a categorical variable or a categorical multiple response set by data of other categorical variables which have the same list of response variants – use [!KO\_DONRECI](#_MACRO_!KO_DONRECI:_CLEANING).
* [!KO\_STRMRC](#_МАКРОС_!KO_STRMRC:_РАЗВЕРТКА) and [!KO\_STRMRD](#_МАКРОС_!KO_STRMRD:_РАЗВЕРТКА) create a ready-to-analyze multiple response set from a string variable containing response entry. [!KO\_ERRDISP](#_МАКРОС_!KO_ERRDISP:_ПРОВЕРКА) is an auxiliary macro.

*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.

**Categorical multiple response set** (MRC) are several variables united by that they jointly are the common depository of data. Data (values) are discrete categories, response variants for a question of multiple (non-alternative) response; so that each respondent would have several different responses entered in their row (case). Each variable of an MRC set thus bears the meaning “one more answering by a respondent”. Cells that remain unfilled by response codes might be left empty or might be filled with a filler code which should be given missing value status in all variables of the MRC set. Usually, an MRC set is undertaken to follow the *regular build*: this is such fill by response codes whereby each next variable appears less and less filled with them; to put it differently, empty/nonvalid cell cannot be found more left than a valid code. The 1st variable is primary therefore. If a respondent did not give answers, “no answer” code is put in the 1st variable.

**Dichotomous or binary multiple response set** (MRD) is the alternative to MRC way to store the same kind of data. In MRD set, variables correspond to categories (response variants or attributes). Variables are as many as there are the variants, and values in the variables are binary: 1 (“yes”, “present”, “selected”) and 0 (“no”, “absent”, “not selected”). This resembles a set of dummy variables, however, unlike dummy variables, MRD set can contain more than one unit in a row, because MRD corresponds to a multiple response question, whereas dummy set corresponds to a single (alternative) response question. In parallel to how in categorical variables they usually provide the code for “no answer to the question”, in MRD they provide the variable “no answer to the question”, which is equal to 1 if all the rest of the set’s variables equal 0, and is equal to 0 otherwise.

# MACRO !KO\_REGMRC: REGULATING A CATEGORICAL MULTIPLE RESPONSE SET

Version 3, May 2012 (Version 1, Feb 2000). Tested on SPSS Statistics 22, 26, 28.

!KO\_regmrc vars= *v1 to v8* /\*Variables constituting an MRC set, should write via “to”

/nvars= /\*Optional: may indicate the number of variables (for speed sake)

/check= YES /\*Optional: screen out duplications of the same code - YES or NO (default);

/\*after YES may list selected codes for such cleaning;

/\*inserting word EXCEPT b/w YES and the codes – clean except these codes

/sole= LEAVE *10* /\*Optional: expel from company (DELETE) or keep isolated (LEAVE) code(s)

/\*incompatible with the rest; list that “sole” codes after the keyword

/random= /\*Optional: randomize code order - YES or NO (default)

/crop= YES /\*Optional: drop spare (empty, w/o valid responses)

/\*variables - NO (default) or YES (incompatible with SELFEXE=NO)

/filler= *0* /\*Optional: filler code assumed in the set (value)

/selfexe= /\*Macro to execute itself (YES, default) or not (NO).

Minimal specification VARS.

The macro imparts *regular* build to a categorical multiple response set (MRC); it is the structure where each subsequent variable in the set is potentially less filled with response codes than the preceding one (see description above). Irregular build usually arises as a result of cleaning or during data entering, and is possible if a respondent has fewer responses than there are variables in the set. The macro does the correction by moving the set’s values from right to left on free places, if there are some. Free places are missing (user- or system-missing) values. This presupposes that if the input MRC uses a filler code, that code has the missing value status there.

Besides lending the regular build, the macro possesses 4 options. (1) You can request to remove code duplicating from the data – of all codes or of specific. (2) You can control that a code incompatible with the rest of codes be the only response by the respondent or got deleted. (3) You can randomize the entry sequence of codes for the respondent – rob it of a systematic flow. (4) You can get rid of extra, containing no valid codes, variables.

The macro does not affect cases having no valid data in input variables VARS.

The macro creates no new variables but alters the data of the input variables themselves. You can make these changes temporary, for one coming procedure: for that, put the macro call under TEMPORARY while s/c SELFEXE=NO.

The macro creates temporary variables with names containing five consecutive symbols *$*, for example, *v$$$$$.\_2*. Therefore, you should better avoid such names in your dataset. If you are running the macro in SELFEXE=NO regime and are creating scratch variables before it, their names should better avoid five consecutive symbols *$*, too.

***Subcommands***

**VARS**

Specify categorical numeric variables constituting an MRC, all of them or just needed. Those must be consecutive variables in the dataset and be specified via “to”: *var* to *var.* You might indicate several adjacent MRC sets as one set, what allows to merge sets (see EXAMPLE 5).

**NVARS**

This unnecessary subcommand can speed up the process of sifting out duplicate codes – i.e., when you’re specifying CHECK=YES. You may want more speed if there are many variables. Indicate in NVARS the number of input variables VARS.

**CHECK**

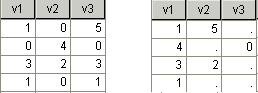
By default/unspecifying and with CHECK=NO, the macro does not sift out repetitions of one and the same valid codes within respondent, if repetitions exist. CHECK=YES sifts out repetitions, so that each valid code will encounter not more than once at output. If there’s the need that this check and removing of duplicates be done only for specific codes rather than for any valid codes, enumerate the needed codes after word YES, e.g.: CHECK=YES 1 3 (only nonnegative values can be in the list). If, on the contrary, you need to ban removing duplicates of specific codes, enumerate them after inserting word EXCEPT, e.g.: CHECK=YES EXCEPT 1 3.

Listing a code more than once doesn’t cause an error: CHECK=YES 1 3 3 is the same as CHECK=YES 1 3.

It is permitted to type, in place of the codes, *variable names* of variables containing them; this opens the possibility to set different codes to different respondents; the codes in these variables need not be nonnegative.

EXAMPLE 1.

!KO\_regmrc vars= v1 to v3 /check= YES.



* MRC had «irregular» build: filler code (herein = 0) is found to the left of valid codes. The macro corrected this by moving the values to the left. Cells coming out free are left empty. To fill them, add subcommand /FILLER=0.
* It was also specified to eliminate code duplicating. A valid value can now be encountered in a case no more than once.

EXAMPLE 2.

!KO\_regmrc vars= v1 to v9 /check= YES 1 3 4 x.

* Duplicates only of codes 1, 3, 4 and X will be removed. X is the name of a numeric variable, and its value could be different for different respondents.

**SOLE**

It is not a rare condition that one or more response variants are by meaning incompatible with all the other response variants. For example, “Don’t know” and “Nothing of the list” are incompatible with other responses as well as with each other. If selected, each of these two variants can be the only response variant by a respondent. SOLE saves the MRC set from violations of the rule. You may prefer: either to delete the code that is incompatible with other codes observed in the respondent, or delete these other, leaving only that one code to the respondent. Use, respectively, keyword DELETE or LEAVE with listing of codes each of which is incompatible with the rest codes. The list may contain any (including negative) numbers presenting valid codes in the data. Variable names cannot be in this list.

For example, specification SOLE= DELETE 99 10 upon finding code 99 or code 10 in conjunction with other codes will delete the code 99 or the code 10 from a respondent. And when the codes 99 and 10 are found in combination with each other, the code 99 of the two will be removed, because it is *earlier* in the list.

Specification SOLE= LEAVE 10 99 upon finding code 10 or code 99 in conjunction with other valid codes will delete those latter. And when the codes 10 and 99 are found in combination with each other, the code 99 will be removed, and 10 will be saved for a respondent, as mentioned *earlier* in the list. Option SOLE itself does not substitute option CHECK: if, for example, code 99 was entered twice in a respondent and option SOLE saves this code for him, it will save both its copies.

EXAMPLE 3.

!KO\_regmrc vars= v2.1 to v2.12 /sole= LEAVE 10 /check= YES 10.

* Code 10 which meant “nothing from the list” is logically incompatible with other valid responses. The investigator bid to delete any other response whatever found in conjunction with code 10.
* To guarantee that code 10 doesn’t repeat in a respondent, the investigator bid to get rid of possible duplicates of that specific code. All other valid codes, if they duplicate, will keep their duplicating.

**RANDOM**

The subcommand (RANDOM=YES) performs randomization[[1]](#footnote-1) of the order in which codes go in a respondent left to right. It can be of use, for example, when the researcher is going to use in their analysis a smaller number of variables than there are in the set, perhaps even a single variable from the set, probably the 1st one, as the representative of the complete set, - and therefore they want to take the distributional profile of responses there closer to that in the complete set. RANDOM=NO and omitting/unspecifying the subcommand – don’t do randomization.

**CROP**

This option (CROP=YES) deletes in the end of the macro’s job “superfluous” variables from the MRC set, the variables containing no valid codes. The s/c is impossible under SELFEXE=NO. By omitting/unspecifying the subcommand and with CROP=NO “superfluous” variables are retained.

**FILLER**

Optional subcommand by which you may indicate the macro a filler code for empty data cells located to the right of valid data. Filler code should be user-missing in the set variables, but the macro won’t give it missing status, assuming that this status is already attached to the code in VARS variables. So, in general it is recommended to indicate, as the filler, the code which already is a filler there with user-missing status. (The code might be any number.) Unspecifying the code as well as omitting the subcommand will leave the cells without valid data empty (system-missing).

**SELFEXE**

By default/unspecifying and with SELFEXE=YES, the macro executes itself. Under SELFEXE=NO, it does not execute itself and waits execution by a further EXECUTE command or a procedure command (reading data) or the next run of the macro (now without SELFEXE=NO). Option SELFEXE=NO gives the possibility to: (1) put the macro in DO IF construct, thereby restricting cases the macro processes; (2) put its run, also accompanied by transformations, under TEMPORARY command, thereby making the effect of the macro temporary – for one next procedure.

With SELFEXE=NO avoid running the macro in succession with *different* variables:

!KO\_regmrc vars= v1 to v8 /selfexe= NO.

!KO\_regmrc vars= w1 to w6.

because you will get erroneous result. In this example, the SELFEXE=NO in the first run is the mistake and should be removed.

If you put the macro call in DO IF construct, a warning may appear, “Command name SET. LOOP has no effect on this command”. Just ignore it.

EXAMPLE 4.

do if region=8.

!KO\_regmrc vars= mr\_a to mr\_f /sole= DELETE 10 /selfexe= NO.

else.

!KO\_regmrc vars= mr\_a to mr\_f /check= YES 10 /selfexe= NO.

end if.

execute.

* Both macro runs give or guarantee the regular build to the MR\_A – MR\_F categorical set.
* For respondents belonging to REGION=8, code 10 is removed from the set if it coexists with other valid codes. For other respondents with valid data, there is provided that code 10 encounter no more than once.
* EXECUTE performs the macro. It is important that both conditioned runs with SELFEXE=NO be busy with the same variables VARS.

EXAMPLE 5. Transient uniting of several consecutive MRC sets.

temporary.

!KO\_regmrc vars= mrc1.1 to mrc3.4 /check= YES /sole= DELETE 99 /selfexe= NO.

select if nvalid(mrc1.1 to mrc3.4)<=5.

tables /mrgroup= $mrc mrc1.1 to mrc1.5 mrc2.1 to mrc2.8 mrc3.1 to mrc3.4 /table= $mrc.

* Three categorical sets with common list of responses, MRC1.1…MRC1.5, MRC2.1…MRC2.8, MRC3.1…MRC3.4, go one after another in the data file. They are mentioned in VARS as a single set.
* The macro gives the unite set the regular build. In order to ward off code duplicating emergent from uniting the sets into one, CHECK is specified; and to ward off uniting of “no answer” code 99 with other codes it is specified to clear that code 99 in such a case.
* The investigator selects to his fancy, with SELECT IF, into table only those respondents who gave up to 5 responses in the united, cleaned by the macro, set.
* Command-procedure TABLES executes the macro and the selection and tabulates the unite set. Because macro (as well as selection) stood under TEMPORARY, the data after the procedure return to their initial state. Macro should have specification /SELFEXE=NO.

***Special regimes***

The macro does not respond to weightedness, filteredness (FILTER, USE), splitness of the dataset. Besides, it turns off weights and split, if CROP=YES. Putting it under TEMPORARY makes sense only with SELFEXE=NO.

# MACRO !KO\_MRDNA: SUPPLYING DICHOTOMOUS MULTIPLE RESPONSE SETS WITH “NO ANSWER” VARIABLE

Version 3, May 2009 (Version 1, Apr 1999). Tested on SPSS Statistics 11, 13, 14, 26.

!KO\_mrdna caps= /\*Prefixes (numbered as there are sets) into "no answer" variable

/separ= *'.'* /\*OR Quoted separator existing in the input variables names;

/\*indicate '' if it is nihil there

/namend= *@* /\*Ending into the "no answer" variable name

/label= *'No answer' 1 'YES' 0 'NO'* /\* Optional: label for "no answer" variable

/\*and, if needed, labels for the values 1 and 0 in it

/format= /\*Format all the set’s variables (F1, ordinal): YES (default) or NO

/regist= NO /\*Register MRD set/s with the "no answer" variable: YES or NO (default);

/\*after YES may suggest quoted label/s for the set/s in their order

/declare= YES /\*Inform if the being created "no answer" variable is already present:

/\*YES or NO (default)

/sets= *v1.1 v1.2 v1.3 / v2.1 to v2.10*

/\*One or more, separated by slash, sets of binary variables

/\*(w/o "no answer" variable); each set either name-by-name or via “to” range.

Minimal specification NAMEND, SETS. SETS must go last.

The macro creates or updates, in multiple response variable sets (MRD), variables meaning “no answer to the question”. This variable is just the “negative” to the sum of all the rest variables of the set: it = 1 where they all = 0, and it = 0 otherwise. The newly created variables will appear in the tail of the dataset: later you will move them to the needed place yourself.

EXAMPLE 1.

!KO\_mrdna separ= '\_' /namend= # /regist= YES

/sets= q12\_1 to q12\_8 /q13.1\_1 to q13.1\_6 /q13.2\_1 to q13.2\_6.

* To the three sets of binary variables, the variable “no answer” will be added, to each. Its name will end by # (and if such variable already exists, it will get updated, overwritten).
* SEPAR informs the macro that in SETS variable names there is separator \_ between the “set name” and the ending individual for the variable. Therefore, the “no answer” variables’ names will be: *Q12#, Q13.1#, Q13.2#.*
* REGIST=YES: the sets will be registered (or re-registered) in the data file by names *Q12, Q13.1, Q13.2* as dichotomous multiple response sets.

***Subcommands***

**SETS**

The subcommand must go last. Indicate here MRD sets for each of which the variable “no answer” is to be created or updated (do *not* indicate here the “no answer” variable itself, if it already exists). Type the set’s variables names, not the registered set’s name. If there are multiple sets, separate them with slash (/). You may write two-mode, and different sets differently: either enumerate the variables name-by-name or indicate the range of consecutive variables in the dataset via “to”. Don’t mix the two modes of entry: you may not write var1 var3 to var10 in this macro.

An MRD set is any collection of numeric binary variables which you are seeing as a “multiple response set”. All the variables must be strictly binary coding: 1 and 0. Empty cells (system-missing) are allowed; “no answer” variable will be system-missing for a case that is system-missing in all the variables of the given set.

**NAMEND**

Ending into the name of “no answer” variable. Specify a symbol or a combination of those. This ending is one for all the sets SETS. “No answer” variable will bear name that is concatenation of the set’s name and the ending. The set’s name is defined by SEPAR subcommand or CAPS subcommand. If in the dataset there already exist variables same-name with the variables the macro is calculating, those variables will be updated.

**CAPS**

Specifying this subcommand is incompatible with specifying SEPAR. In CAPS indicate, for each set in SETS, a different prefix that will become the head in “no answer” variable name of the set. So you specify a list of prefixes, one prefix per set. “No answer” variable name = prefix + NAMEND. The same prefix becomes the set’s name with which it will be registered, if REGIST=YES.

**SEPAR**

Specifying this subcommand is incompatible with specifying CAPS. In SEPAR, the macro is indicated (in quotes/apostrophes) the separator symbol that may be contained in the names of the variables making the sets SETS, and which separates there the common part, or “set’s name”, and the ending individual for the variables. For example, in variable names *MRD\_1 MRD\_2* apparently “\_” serves the separator. If multiple sets are specified in SETS, they all must share the same separator. However, the existence of a separator in the variable names is not necessary.

Actually, the macro does not check whether a set’s variables have identical parts in their names preceding the separator, or whether all or not all the variables have the separator: it looks only at the *first* variable of a set (in the order variables are written in SETS) and isolates its name pre-separator part, which is then taken for the “set’s name”. The “no answer” variable name is compiled of the set’s name, the separator, and the NAMEND.

In more detail, there exist three variants to specify SEPAR:

*Indicate the separator* – suitable if names in SETS contain this separator. Absence of the separator in the name of the 1st (as written in SETS) variable of a set will cause error, with message. Endings of variable names can be not necessarily numbers, but any (e.g., *MRD\_S*). Before the separator, digits might stand in the names (e.g., *MRD1\_1*). “Set’s name” is all the part before the separator. The “no answer” variable name = set’s name + separator + NAMEND.

*Indicate empty quotes/apostrophes (not blank!)* – suitable if endings in the names are integers, while all the preceding, non-digital part of the name is the “set’s name” (e.g., *MRD1*; also not a mistake *MRD\_1*). “Set’s name” is the whole part before the 1st digit. Absence of a digit at the tail of the name of the 1st variable of a set will cause error, with message. The “no answer” variable name = set’s name + NAMEND.

*Omit to indicate (and not specify CAPS)* – suitable if variable names in sets are not compiled as “set’s name” plus ending, but rather are or are accepted by the user as unique. It is the most universal specification of SEPAR. The macro will take the whole name of the 1st variable of a set for the “set’s name”. The “no answer” variable name = set’s name + NAMEND.

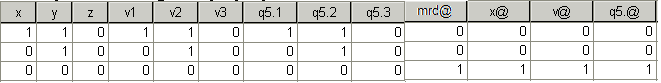
EXAMPLE 2. Situations of different ways to specify SEPAR.

!KO\_mrdna caps= mrd /namend= @ /sets= x y z.

!KO\_mrdna separ= /namend= @ /sets= x y z.

!KO\_mrdna separ= '' /namend= @ /sets= v1 v2 v3.

!KO\_mrdna separ= '.' /namend= @ /sets= q5.1 q5.2 q5.3.



* In the 1st run, the user themselves specifies a name for the set through CAPS. “No answer” variable will be called *MRD@*. SEPAR is not specified.
* In the 2nd run, set *X Y Z* consists of variables without a separator or digits in their names. SEPAR is not specified (or omitted, as CAPS). “No answer” variable name *X@* is formed of the whole name of the 1st variable. This variant would not produce an error with sets *V1 V2 V3* or *Q5.1 Q5.2 Q5.3* either, but is hardly acceptable.
* In the 3rd run, names in set *V1 V2 V3* are compiled of a common alphabetic prefix and individual digital ending. Optimal choice – to specify SEPAR by empty quotes, what will give “no answer” variable *V@*.
* In the 4th run, names in set *Q5.1 Q5.2 Q5.3* have separator symbol (‘.’). Optimal choice – to indicate this separator, what will give “no answer” variable *Q5.@*. Variant SEPAR=‘’ in this instance will give the same result, because endings after the separator are integer numbers.

**LABEL**

Optional subcommand allowing to give a label to the “no answer” variable; you can also set labels to its values (1 and 0). Specify in quotes/apostrophes a label for the variable, for example, ‘No answer’. After it, you may, if you want, to specify both values with their labels, for example: 1 ‘YES’ 0 ‘NO’.

**FORMAT**

By default and with FORMAT=YES, all the variables SETS and computed “no answer” variables are formatted by the macro: F1, ordinal level. Specify FORMAT=NO, if you don’t want that.

**REGIST**

Optional subcommand allowing to “register” in the data file each set of SETS as an MRD set; the “no answer” variable also to enter it. The set will be registered by its “set’s name”. NO – not to register. YES – register; after YES you may indicate, in quotes/apostrophes, descriptive labels for the sets in their order, for example: ‘Which of the following TV shows do you watch?’ ‘What radio programs do you listen?’.

The set name being registered is the one the macro uses in the “no answer” variable name; that is, it is either the name indicated in CAPS or determined according to SEPAR. If the set name ends with symbol “\_” or “.”, the symbol will be dropped from the set name being registered.

**DECLARE**

DECLARE=YES makes SPSS Statistics to message: “Error… A variable with this name is already defined… Execution of this command stops” – in case the being created/updated “no answer” variable is already present in the dataset, that is, it existed before the macro run or has been created by the macro in the current run for an earlier set in SETS. This message plays only a warning role, it does not breaks the job and does not affect the result – the variable will get updated in any case.

By default and DECLARE=NO, the aforesaid message won’t be issued.

***Special regimes***

The macro does not respond to weightedness, filteredness (FILTER, USE), splitness of the dataset. I doesn’t obey temporary (under TEMPORARY) transformations.

# MACRO !KO\_DONRECI: CLEANING OF CATEGORICAL MULTIPLE RESPONSE SET BY OTHER CATEGORICAL VARIABLES WITH THE SAME CATEGORIES

Version 3, Nov 2009 (Version 1, Jun 2001). Tested on SPSS Statistics 20, 22, 25.

!KO\_donreci recip= *v1 to v6* /\*"Recipient", to be cleaned, set of variables where to add or to remove data

/donat= *w1 w2 w3 w4* /\*"Donator", reference, set of variables to collate data with

/values= *1 to* *8* EXCEPT *6 7* /\*List (name-by-name or range via “to”) of non-negative

/\*values-of-interest for adding/removing/recoding

/\*after the range may list, after kw EXCEPT, values to exclude from the range

/clean= ADD /\*Type of cleaning: add to recipients the values absent there but present in

/\*donators (ADD, default);

/\*remove from recipients the values absent in donators (REMOVE);

/\*do this or that by random decision (CHOOSE);

/\*recode in recipients the values absent in donators (RECODE vallist)

/valrand= /\*Randomize values in VALUES list for respondents: YES or NO (default)

/suffic= /\*Optional: add/remove no more than this number of values from VALUES

/\*for a respondent (number or varname)

/report= YES /\*Report, by special variables, numbers of performed additions, removings and

/\*additions failed due to lack of spare places: YES or NO (default)

/replace= *99* /\*Optional: valid values in recipients that you allow to replace

/\*(may use words THRU, LO, HI – as in Recode command)

/bases= /\*Which respondents to be busy with: on intersection of RECIP and DONAT bases

/\*(INTERSECT) or to union the bases (UNITE, default)

/noans= *99* /\*Optional: "no answer" code to insert in the 1st RECIP variable if all RECIP

/\*come out empty in the end

/selfexe= /\*Macro to execute itself (YES, default) or not execute (NO).

Minimal specification RECIP, DONAT, VALUES.

The macro audits (collates-cleans) a categorical multiple response set (MRC) or a single categorical variable by data of other categorical variables with the same categories. A typical context of using the macro is when the being cleaned variables (“recipients”) and the auditor or consultant variables (“donators”) pertain to questionnaire questions having common list of response variants, and one wants to bring the two sides to agreement, so that data in the recipient variables would not contradict data in the donator variables.

The cleaning is, by your choice, enrichment by data, withdrawal of data, and enrichment/withdrawal. One more option is recoding:

1) Enrichment: *donator variables carry in to recipient variables the values absent in the latter*.

2) Withdrawal: *recipients get rid of the values that are extra in comparison with donators*.

3) Enrichment/withdrawal – mixed process when addition, removal, or refrain of the cleaning chaotically alternate during macro run, what leads to *incomplete and random likening of recipients* to donators.

4) Recoding: same as withdrawal (2), but the extra values get recoded into prespecified values rather than are erased.

In all the variants, data in donators are not changed, and the values in recipients that are found also in donators remain in recipients on their places. You can make data changes in recipients temporary, for a single next procedure: for this, put the macro call with subcommand SELFEXE=NO under TEMPORARY command.

The macro creates temporary variables with names containing five consecutive symbols *$*, for example, *v$$$$$.\_2*. Therefore, you should better avoid such names in your dataset. If you are running the macro in SELFEXE=NO regime and are creating scratch variables before it, their names should better avoid five consecutive symbols *$*, too.

EXAMPLE 1.

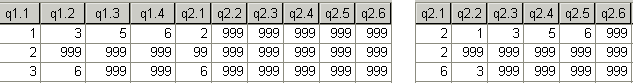
!KO\_donreci recip= q2\_1 to q2\_4 /donat= q1 /values= 2 5 6 7 /clean= REMOVE /report= YES.

!KO\_regmrc vars= q2\_1 to q2\_4.

* Within cases, data are compared – namely, values 2, 5, 6, 7 – between variables Q2\_1 – Q2\_4 (the being cleaned) and variable Q1 (the consultant).
* CLEAN=REMOVE, i.e., if a value is absent in Q1 in the case, it gets erased from Q2\_1 – Q2\_4, if is present there.
* The done removings were counted by the report variables created by s/c REPORT.
* In the situation when variables Q2\_1 – Q2\_4 are a categorical multiple response set, it is reasonable, upon the deletion of a part of data from it, to bring it then to the regular build by ramming the remaining data to the left, – which was done by macro !KO\_REGMRC.

EXAMPLE 2. There were 2 multiple response questions with the same list of response variants. One question was unaided (i.e. without showing the list of variants), “What brands of a shampoo can you name?”, and the other was with prompt “Which more brands, from this card, do you know?”. Answers to the 2nd question were then entered so that they did not include brands checked in the 1st question. Now the researcher decided to do such inclusion, to make the 2nd question the question on general – no care prompted or not – awareness for brands.

!KO\_donreci recip= q2.1 to q2.6 /donat= q1.1 to q1.4 /values= 1 to 6 /replace= 99.



* Both questions are MRC sets, Q1.1 to Q1.4 and Q2.1 to Q2.6, with filler 999 (user-missing). Code 99 meant “no answer”. Possible responses (brands) to the questions: codes 1 to 6.
* By default, the macro performs enrichment: it is requested to copy all such values found in set “Q1” but not found in set “Q2”, to “Q2”. Apart from missing values, it is allowed to replace there the code 99, naturally.
* The result is shown separately on the right. All codes from 1 to 6 found in the 1st set and not found in the 2nd were inserted in the latter. In order all the additions to be done, it was needed, of course, that there be initially enough of “free” (with code 999 or 99) cells; and these happened to be in sufficient.

***Subcommands***

**RECIP**, **DONAT**

In RECIP, specify variables to clean (“recipients”). These are the variables where the macro will add or remove some data. In DONAT, specify consultant variables (“donators”). These are the variables by which to check, and their data won’t be modified. Indicate the variables in the subcommands name-by-name or by range via “to”: *var* to *var* (those then must be adjacent variables in the dataset), but not mixed: *var1 var3* to *var10* is not allowed in this macro. All the variables must be numeric.

Variables in each of these two subcommands can be: 1) either a categorical multiple response set (MRC), full or its part, and the set needs not be registered in the file as an MR set; 2) or a single categorical variable; 3) or multiple self-dependent categorical variables. In RECIP, the variant (3) is permitted only when CLEAN= REMOVE or RECODE.

RECIP and DONAT variables should be categorical (not continuous) and with the same or overlapping domain of values (categories). That is, recipients and donators should contain, for example, responses for a pair of survey questions with the lists of response variants identical or similar.

**VALUES**

RECIP and DONAT variables should be with the same or overlapping domain of values (categories). Type in VALUES, which specifically values you are interested in. It must be nonnegative numbers (the macro doesn’t process negative codes in VALUES). The presence of these values in recipient variables will be collated with their presence in donator variables; and at observing a discrepancy with respect to the value the action CLEAN will be taken.

These values should normally have valid status both in recipient and donator variables. You can specify the list either by enumeration (for example, *1 2 5 16*), or by range via “to”: min to max, for example, *2 to 9*. In the former case the codes need not be integers. When the codes are specified via a range, the macro forms their list as integers from min to max (both these numbers must be integer).

By insertion of keyword EXCEPT after the range you can specify a list of codes that you want to omit from the range. For example, if VALUES= *1 to 5* EXCEPT *2 4*, then the list of codes the macro will generate is *1 3 5*. If no codes follow the word EXCEPT, then the word is ignored.

**CLEAN**

Request cleaning action:

ADD - (also default/unspecifying) addition of values to recipients. Use this option if your goal – **to enrich data of recipients by data of donators**. Each value of VALUES found in the donators and not found in the recipients is inserted in the latter. The value is inserted to the respondent (case) in one copy. Insertion is done into cells containing missings (system- and user-missing values) and, by your desire, in place of specified valid values (see s/c REPLACE). The search for a cell to insert in is done by looking through the variables left to right as they are specified in RECIP. If there were no such cells in the respondent or there are no more such, insertions to the respondent cease. So, if you need that all the VALUES have a chance to be inserted, see to having enough amount of “free” cells in the recipients, what might be done by attaching additional empty variables to RECIP. (Use REPORT subcommand to know the number underinsertions.)

REMOVE - deletion of values from recipients. Use this option if the goal – **to deprive recipients of data lacking in donators**. Each value of VALUES found in the recipients and not found in the donators is erased from the former. All copies of the value will be erased from the respondent (it counts as one removal). With this option, recipient variables need not, generally, constitute an MRC set.

CHOOSE - addition or deletion – by random choice. Use this option in order to **make partly similar data of recipients to data of donators** with respect to VALUES. For each next value of VALUES and each next respondent (case), the decision – to add, to insert or to leave the situation as is[[2]](#footnote-2) – is made randomly and before the action (add or delete) is taken. The result of it becomes that the recipients’ data co-ordinate with the donators not completely, the result depends on random numbers[[3]](#footnote-3), and the saturation of the recipients with data change little in the end. See below about a parameter by CHOOSE.

RECODE *vallist* - **recoding** of values into corresponding other. Each value of VALUES found in the recipients and not found in the donators is replaced by its corresponding value from the *vallist*. All copies of the value are replaced in the respondent. With this option, recipient variables need not, generally, constitute an MRC set. *Vallist* of values must consist of nonnegative values and be of the same length as VALUES list. You may specify it by enumeration or by range – the same ways as you can do in s/c VALUES, and you may use keyword EXCEPT. Values may repeat in *vallist*. You may also type a single value followed by keyword ALL, for example: CLEAN= RECODE 3 ALL. The macro will then recode every value VALUES in that single value (here, 3).

If there wanted is **maximal likening of recipients’ data to the data of donators** in respect to VALUES, run the macro first with option REMOVE, in order to free the recipients from the unwanted values and free cells they occupy, and then run with option ADD, to add the lacking values.

*Parameter by CHOOSE*. You can tune option CHOOSE for predominant addition or predominant deletion. For that, specify number between 0 and 1 after the keyword, e.g.: CLEAN= CHOOSE *0.8*. In this example, additions will have propensity to carry out more often than deletions. CHOOSE *1* is equivalent to CLEAN=ADD. CHOOSE *0* is equivalent to CLEAN=REMOVE. Word CHOOSE without the parameter equals CHOOSE *0.5*. Note that the farther from 0.5 the parameter retreats, the more the ability to leave the situation as is weakens, – in favour of more exhaustive, more assimilating cleaning. Only under parameter close to 0.5 saturation of recipients with data will change little. Instead of a number, you may specify a variable name as the parameter, the variable containing values between 0 and 1, and they may differ for different cases.

EXAMPLE 3. Like EXAMPLE 1, but CLEAN=RECODE.

!KO\_donreci recip= q2\_1 to q2\_4 /donat= q1 /values= 2 5 6 7 /clean= RECODE 22 55 66 77.

* Within cases, data are compared – namely values 2, 5, 6, 7 – between variables Q2\_1 – Q2\_4 (the being cleaned) and variable Q1 (the consultant).
* CLEAN= RECODE 22 55 66 77, i.e. if a value is absent in Q1 in the case, it gets recoded in Q2\_1 – Q2\_4 as follows: 2 into 22, 5 into 55, 6 into 66, 7 into 77.

**SUFFIC**

This is the limit for the number of allowed added or removed values VALUES for a respondent (dataset case). By default, it equals the number of values in VALUES list, – i.e., the macro attempts to insert (withdraw) into recipients, where it is needed, all the values of the list. In SUFFIC, you may lay a limitation for the number of insertions (withdrawals) by indicating a positive number. Then the macro, having added (removed) that number of values to a respondent, ceases business with the respondent.

Instead of a number, you may indicate a *variable* containing nonnegative numbers; those can be different for different respondents. The macro internally rounds data in this variable and treats missings in it as 0. Value 0 means that no insertions no removals will take place for the respondent.

Along with SUFFIC there usually makes sense to use VALRAND=YES, in order to equalize chances of different VALUES values for insertion (removal) under the condition of limitation for the number of insertions (removals).

SUFFIC is ignored when CLEAN=RECODE.

EXAMPLE 4. Same as EXAMPLE 2 but restricting the number of additions.

!KO\_donreci recip= q2.1 to q2.6 /donat= q1.1 to q1.4 /values= 1 to 6 /replace= 99 /suffic= 2.

* It is allowed to insert no more than 2 values for each respondent, and in so doing – see EXAMPLE 1 – the 1st respondent will get added values 1 and 3 only, but not 5 and 6.

**VALRAND**

By default/unspecifying and with VALRAND=NO, the macro processes the values in the order they go in VALUES subcommand. Option VALRAND=YES randomizes the order in that list differently for different respondents; so first to be added (deleted) will pretend once these, once those of the values. This option makes sense mostly in conjunction with specified SUFFIC or when the amount of free places to insert in is knowingly less than it is required to insert the data.

VALRAND=YES is ignored when CLEAN=RECODE.

EXAMPLE 5. Copy for each respondent up to 10% randomly selected out of his/her responses.

compute suffic= 0.1\*nvalid(var1 to var50).

!KO\_donreci recip= x1 to x16 /donat= var1 to var50 /values= 1 to 75 /suffic= suffic /valrand= YES.

* Variables to copy values to – X1 to X16. There will be inserted, into cells without valid data, values absent in those variables but present in variables VAR1 to VAR50. Possible values to collate-and-insert are the codes 1 to 75.
* No more codes can be inserted than variable SUFFIC indicates, what was computed as 10% of the amount of valid responses a respondent gave in VAR1 to VAR50.
* Without VALRAND=YES, only first codes in VALUES (1, 2, etc.) could have the chance to be inserted, but now all codes from 1 to 75 have equal chance.

**REPLACE**

This option plays a role when CLEAN= ADD or CHOOSE. It permits to mark which cells in recipients – in addition to missing (system- and user-) values – are to be considered “free” for insertion. Specify one or more numeric values with valid status in the recipient variables. The values can be any, including decimal fractional or negative. You may use keywords allowed in SPSS command RECODE: THRU, LO, HI. For instance, REPLACE= LO THRU 10 99 means: permitted to insert in a cell that contains either a missing value (that’s always) or a value not greater than 10 or the value 99.

EXAMPLE 6. There was a multiple response question, “Which TV channels do you watch?” (Q1) and then another with single response, “Which of the channels do you watch most?” (Q2). Code 99 fastened for “no answer”. The task is to make agree of each respondent between the questions.

!KO\_donreci recip= q1.1 to q1.20 /donat= q2 /values= 1 to 15 /replace= 99.

!KO\_donreci donat= q1.1 to q1.20 /recip= q2 /values= 1 to 15 /replace= 99 /valrand= YES.

* The 1st macro run adds any channel (of the 15 ones) mentioned in Q2 but not in “Q1”, into “Q1” (MRC set Q1.1 to Q1.20). Apart from missing data, code 99 is allowed to replace there.
* The 2nd run performs the back cleaning, Q2 by “Q1”, because the researcher decided to get rid, if possible, of no-answers in Q2. Because in “Q1” there can be several channels mentioned, VALRAND selects randomly which of them to insert in Q2 on the place of “no answer”.
* A researcher less inclined to simulation would have preferred another approach: see EXAMPLE 7.

**BASES**

This subcommand defines which dataset cases (respondents) the macro will be busy with.

UNITE - (also default/unspecifying) cases which have any valid values in at least in one of the two variable sets, RECIP or DONAT. In other words, the respondent bases will be merged.

INTERSECT - cases which have any valid values in both sets, RECIP and DONAT. In other words, incoincidence of the respondent bases will be recognized and preserved.

“Any valid values” implies: not just from VALUES list.

Thus, if RECIP and DONAT correspond to two survey questions (with similar response variants) and the two questions interrogated not completely the same subsamples, UNITE will affect by cleaning the union of the subsamples, i.e., will merge them, but INTERSECT will affect by cleaning only the intersection of the subsamples. If both respondent subsamples coincide, the subcommand plays no role.

**REPORT**

You can request (REPORT=YES) the report about the done, for each respondent (case), removals and insertions of values into the recipient variables. Three count variables are created or updated: REMOVE\_ (number of deleted values or, with CLEAN=RECODE, number of recoded values), ADD\_ (number of inserted values), ADDFAIL\_ (number of values which could not be inserted due to lack of “free” cells in the recipients). The first one is full of zeros under CLEAN=ADD, and the other two – under CLEAN=REMOVE. All the three variables will be empty (system-missing) for respondents who were not busy by the macro. (Respondents processed by the macro are defined by BASES subcommand and also by some DO IF condition inside which the user might put the macro call – see SELFEXE subcommand.) If SUFFIC is specified, the sum of the three counter variables is always less or equal to the value of SUFFIC.

With CLEAN=CHOOSE, because result depends on random numbers, data of the counters change from run to run with the same input. In CLEAN=CHOOSE situation, sometimes it is possible to face the picture where a respondent has free cells in recipients on exit from the macro and yet, according to variable ADDFAIL\_, not all insertions were made to him. That is possible, because insertion-removal CHOOSE goes dynamically, and free places to insert into did not happen in a moment *needed*.

**NOANS**

You may specify a code (any number) that you will want to insert into the *first* one of RECIP variables when a respondent is found to have not a single valid value in the recipients on exit from the macro. This code has the meaning of “no answer”; specifying it guarantees that recipient variables will not leave without valid data for respondents the macro was busy with (the code won’t be inserted to respondents not processed by the macro; and the code’s insertion is not counted by the variables created by REPORT).

Instead of a code you may indicate a *variable* with values; thus, different respondents might be appointed different NOANS codes.

**SELFEXE**

By default/unspecifying and with SELFEXE=YES, the macro executes itself. Under SELFEXE=NO, it does not execute itself and waits execution by a further EXECUTE command or a procedure command (reading data) or the next run of the macro (now without SELFEXE=NO). Option SELFEXE=NO gives the possibility to: (1) put the macro in DO IF construct, thereby restricting cases the macro processes; (2) put its run, also accompanied by transformations, under TEMPORARY command, thereby making the effect of the macro temporary – for one next procedure.

Option SELFEXE=NO has the following limitation. If you are running the macro in succession with it, and in *both* runs VALRAND=YES, VALUES in the 2nd run must be the *same* as in the 1st, - otherwise wrong result will occur. I.e., avoid the following case:

!KO\_donreci recip= … /donat= … /values= 2 3 4 5 /valrand= YES /selfexe=NO.

!KO\_donreci recip= … /donat= … /values= 1 to 10 /valrand= YES.

It would be correct to remove SELFEXE=NO from the first run.

One should also keep in mind that subcommands CLEAN=CHOOSE and VALRAND=YES generate random numbers; these in two successive runs of the macro with SELFEXE=NO will be numbers different from random numbers generated in two successive runs of the macro without SELFEXE=NO. So results won’t be the same.

If you put the macro call in DO IF construct, a warning may appear, “Command name SET [or: ECHO]. LOOP has no effect on this command”. Just ignore it.

EXAMPLE 7. Same situation as in EXAMPLE 6.

!KO\_donreci recip= q1.1 to q1.20 /donat= q2 /values= 1 to 15 /replace= 99.

do if nvalid(q1.1 to q1.20)=1.

!KO\_donreci donat= q1.1 to q1.20 /recip= q2 /values= 1 to 15 /replace= 99 /selfexe= NO.

end if.

execute.

* The 1st macro run is identical to that of EXAMPLE 6: inserts a wanting response from Q2 into set “Q1”.
* The 2nd run inserts a response from set “Q1” into Q2 replacing “no answer” (99) or emptiness in that variable; but this is being done only when the response is the only valid response in “Q1” set. Because otherwise – if there are several responses – it’s unclear which of them to take, and so then let Q2 better remain with “no answer”.
* EXECUTE executes the 2nd macro run. You could have specified SELFEXE=NO in the 1st run as well: EXECUTE will carry out both runs.

EXAMPLE 8.

do if region=8.

!KO\_donreci recip= q1 /donat= q2.1 q2.2 /values= 0 2 /replace= -99 /selfexe= NO.

else.

!KO\_donreci recip= q1 /donat= q3\_1 q3\_2 /values= 1 4 /replace= 0 99 /selfexe= NO.

end if.

execute.

* For respondents belonging to region 8 (this is the 1st macro run), variable Q1 is being cleaned by variables Q2.1, Q2.2, and the cleaning is the insertion of failing values 0 or 2 in place of value -99 (or in place of a missing).
* For other respondents (the 2nd macro run), variable Q1 is being cleaned by variables Q3\_1, Q3\_2, and the cleaning is the insertion of failing values 1 or 4 in place of value 0 or 99 (or in place of a missing).
* To save time, both runs are executed at one time by external command EXECUTE; they don’t execute themselves (SELFEXE=NO in both).

***Special regimes***

The macro does not respond to weightedness, filteredness (FILTER, USE), splitness of the dataset. Putting it under TEMPORARY makes sense only with SELFEXE=NO.

***Some questions***

*What’s the difference: (i) to put the macro run into DO IF or (ii) to indicate in s/c SUFFIC the variable with certain values zero?* These are real alternative ways to solve the same task: to exclude some cases. But alternatives not fully equivalent. DO IF excludes some cases from processing entirely, and SUFFIC with zero values excludes such cases from the chief action of the macro (subcommand CLEAN) but not from effects of subcommands BASES, NOANS, REPORT. Besides, CLEAN=RECODE ignores s/c SUFFIC.

# МАКРОС !KO\_STRMRC: DEPLOYING OF STRING VARIABLE INTO CATEGORICAL MULTIPLE RESPONSE SET

Version 2, Jul 2004 (Version 1, Feb 2001). Tested on SPSS 11, 11.5, 13, 14.

!KO\_strmrc strvar= v1 /\*Текстовая переменная с ответами на вопрос множественного ответа

/pack= ONECHAR /\*Как набиты в ней коды: одно-значные впритык (ONECHAR);

/\*одно- и дву-значные отдельно те от других (DISPART);

/\*коды любой ширины через разграничитель (FREE)

/check= YES /\*Отсеивать возможные повторения того же кода (YES, тж п/у) или не отсеивать (NO)

/eqcase= YES /\*Если в STRVAR есть буквы, то считать ли строчные и заглавные буквы

/\*одним и тем же (YES, тж п/у) или не считать (NO)

/categs= 1 2 3 4 a b c /\*Коды ответов, использованные в переменной STRVAR, все или нужные;

/\*Если eqcase=NO, то значимо, строчными или заглавными писать в этом списке буквы;

/\*Если все коды - целые неотр числа, список можно указать диапазоном ч-з to: min to max

/\*и тогда после сл EXCEPT можно перечислить коды для выпуска из диапазона

/codes= 1 2 3 4 10 11 12 /\*Соответствующие CATEGS числовые коды (неотриц числа) в создаваемые

/\*переменные, либо SAME – взять сам список CATEG;

/\*Если опустить CODES, переменные выйдут текстовыми

/filler= 0 /\*Опционально: код-заполнитель для создаваемых переменных (он станет user-missing)

/noans= 99 /\*Опционально: код «нет ответа на вопрос» для создаваемых переменных

/limit= /\*Опционально: максимальное число ответов, какое разрешено было сделать респонденту;

/\*столько будет переменных в наборе

/separ= '.' /\*Опционально: окавыченно разделитель в имена создаваемых переменных

/regist= YES /\*Регистрировать ли созданный MRC набор: YES или NO (тж п/у);

/\*после YES можно указать окавыченный ярлык для набора.

Минимум надо задать STRVAR, PACK, CATEGS.

Данные, относящиеся к одному вопросу на множественный (неальтернативный) выбор, бывает удобно набивать в единую текстовую переменную. Макрос создаст из подобной текстовой переменной готовый к анализу категориальный набор множественного ответа (MRC). Можно сделать переменные числовыми или текстовыми, принять или отсеять повторение ответов.

***Способы набивки в текстовую переменную***

Текстовая переменная может быть набита одним из трех способов:

* Набивка *свободного* типа: коды ответов могут быть произвольной, в том числе разной, ширины (данный макрос разрешает ширину кода до 8 символов). Коды набиваются через разграничитель (например, пробел или плюс). Пример: **12 563 G6h R%#** - здесь набито через пробел 4 некоторых кода.
* Набивка типа *односимвольные коды впритык*: код может состоять только из одного символа, и набиваются коды без разграничения: **3a68S\*8** – здесь набито 7 кодов (код 8 набит дважды).
* Набивка *разделенного* типа: коды могут быть шириной в 1 или 2 символа (по кр. мере, данный макрос разрешает до 2-х символов). Все односимвольные коды идут в строке прежде двусимвольных, а между теми и другими стоит разграничитель. Пример: **36+121321** – здесь набиты коды 3, 6, 12, 13, 21. Если односимвольных кодов нет – например респондент не дал таких ответов – а есть только двусимвольные, разграничитель все равно должен присутствовать перед ними: **+121321**.

При любом типе набивки символы, из которых состоят коды, могут быть любыми значками, кроме тех что приняты за разграничители. В том случае если порядок ответов респондента важен (например, когда вопрос просил *выбрать и проранжировать* ответы), набивка разделенного типа, очевидно, не годится, т.к. в ней коды сгруппированы по их ширине[[4]](#footnote-4). С другой стороны, этот тип – самая быстрая и удобная набивка обыкновенного вопроса на множественный ответ. В самом деле: можно набивать числовые коды (типично ответы в анкетах кодируют числами 1, 2, 3, и т.д.; максимум тут можно будет использовать код 99). Также, в этой набивке не нужно вводить разграничитель – т.е. нажимать еще клавишу – после каждого кода. Эта набивка может быть даже лучше чем применение «автососкока», существующего в специальных программах по ручному вводу данных, т.к. автососкок хорош лишь тогда, когда все коды одной ширины. Впрочем, разделенная набивка требует внимательности или навыка от набивщика. Макрос !KO\_ERRDISP (см. ниже) может обнаружить ошибки, допускаемые набивщиками в разделенной набивке.

***Подкоманды***

**STRVAR**

Имя текстовой переменной, содержащей ответы на вопрос. Переменные создаваемого набора будут поименованы как имя переменной STRVAR целиком + разделитель SEPAR (если задан) + порядковый номер создаваемой переменной от 1 до LIMIT (см.). Т.о., если вы используете SPSS ниже 12-й версии – где имя переменной может быть до 8 символов – имя STRVAR сделайте достаточно коротким, чтобы все переменные могли получиться. Создаваемые переменные должны несовпадать именами с какими-л. существующими в файле переменными.

**PACK**

Укажите тип набивки в текстовой переменной:

FREE – коды разделены (между ними разграничитель). Коды могут быть шириной до 8 символов и состоять из любых символов.

ONECHAR – коды идут впритык друг к другу. Коды могут быть любыми символами единичной ширины.

DISPART – коды идут впритык друг к другу. Коды могут быть шириной до 2 символов и состоять из любых символов (типично использование одно- и дву-значных натуральных чисел). Двусимвольные коды должны идти после односимвольных. Перед первым набитым двусимвольным кодом (независимо от того, есть ли перед ним односимвольные коды) должен стоять разграничитель.

Разграничителями макрос считает *пробел* и, кроме того, символы, какие указаны в первой команде тела макроса, DEFINE, в аргументе PM…!DEFAULT. Сейчас там указан *плюс*. Значит, макрос считает разграничителями пробел и плюс. Вы можете установить там свои символы, указав их один за другим, впритык. Например, (‘.,;’) будет значить, что разграничителем считается, кроме пробела, любое из: точка, запятая, точка-с-запятой. Если хотите иметь разграничителем только пробел, уберите все находящееся между апострофами: (‘’)[[5]](#footnote-5).

Вереница символов-разграничителей в STRVAR допустима; т.е. напр., ‘+++’ будет считаться тем же что ‘+’. Встреча разграничителей в набивке ONECHAR не ведет к ошибке: ’25 A C7’ считается как ‘25AC7’. В набивке DISPART разграничитель (или их вереница) может, по правилам, встретиться не более одного раза: т.е. набивка ‘13+1422+25’ является ошибкой набивки (макрос *не сообщает* о ней).

**CATEGS**

Здесь надо указать коды ответов (все или нужные), использованные в переменной STRVAR. Это могут быть числа, буквы, прочие символы[[6]](#footnote-6), их сочетания – все что позволяет тип набивки, PACK. В случае, если данные набиты целыми неотрицательными числами, то вместо их перечисления можно указать диапазон min to max, например 1 to 20. Макрос сам составит список из чисел от min до max. В SPSS ниже 13-й версии длина списка CATEGS должна быть меньше 255 символов, включая пустоты между ними, иначе SPSS сообщит об ошибке[[7]](#footnote-7).

Если CATEGS заданы диапазоном min to max, то дальше можно, вставив слово EXCEPT, привести поименно коды, которые хотите выпустить из диапазона, - они будут исключены макросом из формируемого списка кодов. Например, CATEGS= 1 to 5 EXCEPT 2 4. В итоге список будет: 1 3 5.

**CODES**

Этот список – соответственный списку CATEGS – задает коды, которые станут значениями переменных MRC-набора. Если требуется создать переменные числовыми, необходимо либо привести поименный список неотрицательных чисел в количестве, сколько кодов в CATEGS, либо указать CODES=SAME, что означает: использовать сам список CATEGS. В последнем случае CATEGS должен состоять целиком из неотрицательных чисел. Для того же, чтобы создать переменные текстовыми, надо умолчать или незадать подкоманду CODES. Значениями переменных при этом станут сами коды CATEGS.

**CHECK**

По умолчанию/незаданию этой подкоманды и при CHECK=YES макрос следит, не повторяются ли одни и те же коды в строке переменной STRVAR, и, если повторяются, не учитывает повторные встречи кодов. В выходящем наборе один и тот же код будет встречаться не более раза. При CHECK=NO макрос допускает повторения кодов в выходящие переменные, если коды повторяются в текстовой переменной.

**EQCASE**

Эта подкоманда играет роль, если в CATEGS присутствуют буквы. По умолчанию/незаданию и при EQCASE=YES регистр букв игнорируется в текстовой переменной: код *Abc*, например, будет считаться тем же что код *ABC*; и неважно, в каком регистре вы напишете буквы в CATEGS. При EQCASE=NO регист учитывается. В этом случае проследите, чтобы коды в списке CATEGS отвечали регистром букв нужным вам кодам, набитым в текстовой переменной.

**SEPAR**

Это необязательный разделитель в имена выходящих переменных. Укажите символ в кавычках или апострофах.

**LIMIT**

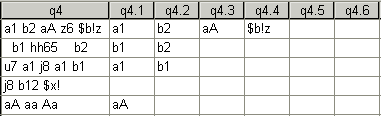
Заказ, сколько создать переменных в наборе. По умолчанию создается столько переменных, сколько кодов в CATEGS. Если респондентам разрешено было сделать не более стольких-то ответов, вы можете указать этот предел здесь: столько и выйдет переменных. С другой стороны, если было разрешено выбрать каждый вариант более раза, и следовательно вы допускаете повторение кодов (CHECK=NO), то вам может понадобиться заказать LIMIT больше, чем число кодов CATEGS, чтобы переменных создалось достаточно для размещения всех ответов.

**FILLER, NOANS**

FILLER это код-наполнитель для MRC-набора. Вы можете указать любое число (или не число, если переменные создаете текстовыми). Код-наполнитель будет иметь статус пропущенного значения в во всех переменных набора или, если это тот же код что NOANS, во всех переменных кроме первой. NOANS это код «нет ответа». Он встретится только в первой переменной набора – у респондентов, не давших ни одного ответа из списка CATEGS. Обе подкоманды необязательны. Не указывайте в качестве FILLER или NOANS коды из CATEGS или CODES.

ПРИМЕР 1. Набивка свободного типа.

!KO\_strmrc strvar= q4 /pack= FREE /check= YES /eqcase= NO /categs= a1 a2 b1 b2 aA $b!z /separ= '.'.

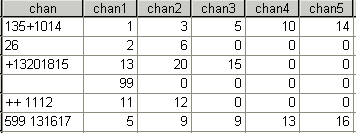


* Переменная Q4 распакована в набор из 6-ти переменных (т.к. кодов CATEGS шесть, а п/к LIMIT не задана). Переменные вышли текстовыми, т.к. п/к CODES не задана. Интересовавшие пользователя коды указаны в CATEGS. Они, если найдены в Q4, перешли в созданные переменные.
* Т.к. CHECK=YES, код “a1”, встречающийся дважды у 3-го респондента, взят в набор только один раз. Т.к. EQCASE=NO, макрос уважает регистр букв: например, коды “aA”, “aa” и “Aa” это для него не одно и то же.

ПРИМЕР 2. Набивка разделенного типа.

!KO\_strmrc strvar= chan /pack= DISPART /check= NO /categs= 1 to 20 EXCEPT 18 19 /codes= SAME /limit= 5

/filler= 0 /noans= 99 /regist= YES 'Какие каналы вы смотрите больше всего?'.



* Затребованные для распаковки коды CATEGS – натуральные числа от 1 до 20, кроме 18 и 19. В выходящих числовых переменных набора, CHAN1 – CHAN5, они будут представлены самими собой (CODES=SAME).
* В качестве наполнителя (пропущенное значение) положен код 0, а в качестве «нет ответа на вопрос» (точнее, «не найдено ничего из CATEGS») код 99.
* Повторение подряд символов-разграничителей у 5-го респондента (плюс плюс пробел) не ведет к неправильной распаковке: ошибкой было бы набить их в разных местах строки.
* Т.к. CHECK=NO, оба кода 9 (6-й респондент) взяты в выходящий набор переменных. Т.к. заказано было создать именно 5 переменных (LIMIT=5), код 17 у этого респондента остался не взятым – не хватило переменных.

**REGIST**

Необязательная подкоманда, позволяющая «зарегистрировать» в файле данных созданный набор. Зарегистрирован он будет под именем переменной STRVAR. Опция регистрации существует в SPSS 11.5 и выше с установленным модулем Custom tables. NO (тж п/у) – не регистрировать. YES – зарегистрировать; после YES можно указать в кавычках/апострофах ярлык набора (формулировку вопроса), например: ‘Какие из перечисленных телепередач вы смотрите?’.

***Особые режимы***

Макрос бессмысленно пускать под командой TEMPORARY. Он не реагирует на взвешенность, фильтрованность, расщепленность файла данных.

# МАКРОС !KO\_STRMRD: DEPLOYING OF STRING VARIABLE INTO DICHOTOMOUS MULTIPLE RESPONSE SET

Version 2, Sep 2004 (Version 1, Aug 2000). Tested on SPSS 11, 11.5, 13, 14.

!KO\_strmrd strvar= v1 /\*Текстовая переменная с ответами на вопрос множественного ответа

/pack= FREE /\*Как набиты в ней коды: одно-значные впритык (ONECHAR);

/\*одно- и дву-значные отдельно те от других (DISPART);

/\*коды любой ширины через разграничитель (FREE)

/eqcase= YES /\*Если в STRVAR есть буквы, то считать ли строчные и заглавные буквы

/\*одним и тем же (YES, тж п/у) или не считать (NO)

/categs= 1 AB a22 7&,5 !w:^ cd /\*Коды ответов, использованные в переменной STRVAR, все или нужные;

/\*Если eqcase=NO, то значимо, строчными или заглавными писать в этом списке буквы;

/\*Если все коды - целые неотр числа, список можно указать диапазоном ч-з to: min to max

/\*и тогда после сл EXCEPT можно перечислить коды для выпуска из диапазона

/indxs= 1 2 3 4 5 6 /\*Cоответствующий списку categs cписок окончаний (числа, буквы, и т п) в имена

/\*создаваемых переменных; или SAME - использовать сам список categs

/noans= @ /\*Окончание в имя переменной НЕТ ОТВЕТА; если она не нужна, не задавайте

/separ= '.' /\*Опционально: окавыченно разделитель в имена создаваемых переменных

/regist= YES /\*(SPSS 11.5 и выше): Регистрировать ли созданный MRD набор: YES или NO (тж п/у);

/\*после YES можно указать окавыченный ярлык для набора.

Минимум надо задать STRVAR, PACK, CATEGS, INDXS.

Данные, относящиеся к одному вопросу на множественный (неальтернативный) выбор, бывает удобно набивать в единую текстовую переменную. Макрос создаст из подобной текстовой переменной готовый к анализу двоичный, или дихотомический, набор множественного ответа (MRD). О способах набивки кодов ответов в текстовую переменную см. выше, в описании макроса !KO\_STRMRC. Переменных, одноименных создаваемым, должно не быть во входящем файле.

Макрос создает на каждый вариант ответа одну числовую переменную. Если вопрос на множественный выбор разрешал выбирать один и тот же ответ более раза и вы хотите иметь соответственно по нескольку двоичных переменных на каждый вариант ответа, воспользуйтесь !KO\_STRMRC и затем переделайте категориальный набор в двоичный с помощью !KO\_MRCMRD (“Categorical – Binary recodings”).

***Подкоманды***

Подкоманды STRVAR, PACK, CATEGS, EQCASE, SEPAR, REGIST тождественны таковым макроса [!KO\_STRMRC](#_МАКРОС_!STRMRC:_РАЗВЕРТКА).

**INDXS**

Укажите поименный список окончаний в имена создаваемых переменных. Этот список соответствен списку кодов CATEGS. В качестве окончаний можно использовать числа, буквы, иные возможные в именах символы или их сочетания. Не используйте числобуквы (1a 3b и т.п.), т.к. они не считаются макросом цельными «словами». Буквочисла (a1 b3) – можно. Вместо списка можно указать ключевое слово SAME, т.е. – использовать сам список CATEGS. Тогда последний должен быть из цельных «слов», т.е. например чисел, букв(осочетаний) или буквочисел.

ПРИМЕР 1.

!KO\_strmrd strvar= chan /pack= DISPART /categs= 1 to 20 EXCEPT 18 19 /indxs= SAME /noans= na.

* Создастся 18 двоичных переменных с именами CHAN1, CHAN2,…, CHAN17, CHAN20.
* А также 19-я переменная набора, «нет ответа», CHANNA.

**NOANS**

Эта необязательная подкоманда заказывает создать к набору переменную «нет ответа на вопрос», точнее, «не выбран ни один код из CATEGS». Укажите символ или символосочетание, на что должна оканчиваться такая переменная. Переменная создается как негатив остальных двоичных переменных: она равна 1, если они все равны 0, и равна 0 в противном случае.

***Особые режимы***

Макрос бессмысленно пускать под командой TEMPORARY. Он не реагирует на взвешенность, фильтрованность, расщепленность файла данных.

# МАКРОС !KO\_ERRDISP: CHECKING OF DATA ENTRY OF DISPART TYPE

Version 1, Dec 2003. Tested on SPSS 11, 11.5, 13, 14.

!KO\_errdisp vars= v1 v2 /\*Список (поименно) текстовых переменных с набивкой типа dispart

/id= /\*Опционально: одна или более переменных-идентификаторов наблюдений.

Минимум надо задать VARS

Этот подсобный макрос проверяет текстовые переменные, набитые *разделенным* способом (см. в описании макроса !KO\_STRMRC выше), на присутствие некоторых ошибок набивки. Макрос выдает по каждой из указанных текстовых переменных номера анкет (респондентов) где встречена ошибка хотя бы одного из трех видов:

* число цифр (вообще символов) среди дву-значных кодов нечетное, например: 3+10125 – здесь, предположительно, хотели набить коды 3, 10, 12, 15, но единицу в «15» забыли.
* знак-разграничитель встречен более 1 раза в разных местах строки, например, 3+1012+15.
* в отсутствие в строке разграничителя цифра (вообще символ) встречается более одного раза, что – в случае если повторения кодов не предусматривались анкетой – скорее всего означает, что забыт разграничитель или что одно-значный код повторен.

Макрос не считает ошибкой, если строка оканчивается на разграничитель или если разграничитель идет вереницей (повторяется подряд). Макрос считает разграничителем пробел (всегда) и, кроме того, плюс. Вы можете заменить плюс на любой другой символ или символы в качестве разграничителей. Для этого укажите их в макросе, в команде DEFINE, аргументе PM…!DEFAULT(‘’). Например, (‘;,’) будет значить, что разграничителями считаются, кроме пробела, точка-с-запятой или запятая[[8]](#footnote-8).

***Подкоманды***

**VARS**

Укажите поименно одну или более текстовых переменных, набитых кодами *разделенным* способом.

**ID**

Вы можете указать одну или более переменных, идентифицирующих респондентов. По умолчанию используется номер наблюдения в файле.

1. Managing of random number seed in SPSS Statistics: menu Transform – Random Number Generator. [↑](#footnote-ref-1)
2. Decision to “leave the situation as is” signifies the occasion when (a) the decision to “delete” a value from a respondent was randomly chosen, and then it turns the action isn’t feasible (because condition “the value is present in the recipients but not in the donators” is false); or the occasion when (b) the decision to “add” a value to a respondent was randomly chosen, and then it turns the action isn’t feasible (because condition “the value is absent in the recipients but not in the donators” is false). Notice that in situation (a) the opposite action, to add, might sometimes be carried out instead – but it will be skipped; likewise in situation (b) the opposite action, to delete, might sometimes be carried out instead – but it will be skipped. [↑](#footnote-ref-2)
3. Managing of random number seed in SPSS Statistics: menu Transform – Random Number Generator. [↑](#footnote-ref-3)
4. Но порядок кодов внутри группы односимвольных и внутри группы двусимвольных может быть произвольным. [↑](#footnote-ref-4)
5. Нюанс касательно разграничителя-точки такой: не указывайте ее в одиночестве (‘.’) или последней (‘,;.’), - это приведет к неправильной работе. Можно продублировать точку (‘..’) или указать не последней (‘.,;’). [↑](#footnote-ref-5)
6. Макрос не разрешает использовать в коде ответа слэш (/). [↑](#footnote-ref-6)
7. Следовательно, при использвании тех версий нужно заранее позаботиться, чтобы текстовая переменная, набитая по свободному (FREE) способу, употребляла как можно более узкие коды, если кодов в CATEGS предполагается указать много. [↑](#footnote-ref-7)
8. Нюанс касательно разграничителя-точки такой: не указывайте ее в одиночестве (‘.’) или последней (‘,;.’), - это приведет к неправильной работе. Можно продублировать точку (‘..’) или указать не последней (‘.,;’). [↑](#footnote-ref-8)