***Clustering tendency***

SPSS macros by Kirill Orlov

kior@akado.ru, ttnphns@gmail.com

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

All rights reserved

*Clustering tendency*. Some methods of exploration which allow to judge preliminary before cluster analysis whether there are clusters in the data. Block-diagonalization of a distance matrix between objects can suggest how many clusters there are. Hopkins statistic is based on simulations of random no-cluster data and comparison of these with the observed data.

*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.

* [Block-diagonalization](#_MACRO_!KO_BLOCKDIAG:_BLOCK-DIAGONAL) needs a distance matrix. This method has an implicit kinship with hierarchical clustering by the single linkage method.
* [Hopkins statistic](#_MACRO_!KO_HOPKINS:_HOPKINS) needs “cases - quantitative features” data. And assumes Euclidean space and Euclidean distances. Hopkins statistic poses a number of conditions, but it is not tied implicitly with some method of clustering.

# MACRO !KO\_BLOCKDIAG: BLOCK-DIAGONALIZATION OF DISTANCE MATRIX

Version 1, Mar 2023. Tested on SPSS Statistics 22, 27, 30.

!KO\_blockdiag matrix= *VAR1 to VAR80* /\*Columns constituting the matrix body (may use "to")

/id= /\*Optionally: numeric identifier variable of cases (rows)

/method= VAT /\*Method: VAT (default) or MDS

/poster= YES /\*"Posterization" after permutation: YES or NO (default)

/plot= /\*Heatmap: GREY (default) or RGREY;

/\*word LABEL can be appended then; or NONE

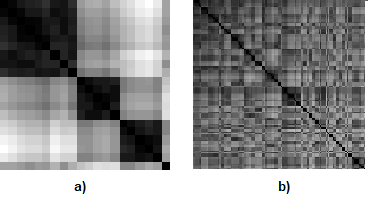
/bounds= /\*Bounds for the heat scale on the map: AUTO (default),

/\*OBS, or min max

/mds= /\*With METHOD=MDS: transformation in MDS: SPLINE (default), ORDINAL, INTERVAL.

Minimal specification MATRIX.

Let there be a matrix of distances (dissimilarities) between objects. Block-diagonalization of the square symmetric distance matrix is the reordering of its rows/columns – i.e., the objects – so that objects with small distances between them become placed next of each other in the matrix (i.e., their indices become close). On the heatmap, such permuted matrix looks consisting of diagonal blocks. It will be the more distinctly, contrastly diagonal-blocked the stronger there exists the tendency towards clusteredness among the objects, that is, the stronger the distances fall themselves apart into “within-cluster” (small) and “between-cluster” (large). Therefore, one can approximately judge by the heatmap about presence or absence of clusters in the data and about the number of clusters, without doing cluster analysis. Each cluster on the heatmap after block-diagonalization of the matrix looks like a block on the diagonal. If there are no clusters in the data, block-diagonalization will not show clear blocks on the heatmap.



**Fig. 1**. Heatmap of a block-diagonalized distance matrix: a) with clear clusters in data, b) without clusters in data.

The macro does block-diagonalization (permutes rows/columns) of the input distance matrix, saves the obtained matrix as a new unnamed dataset and draws heatmap.

**Algorithm**

With METHOD=VAT, the macro does reordering of rows/columns of the distance matrix by VAT algorithm. VAT algorithm (“Visual Assessment of [Cluster] Tendency”) is described detailed in [1], also in [2,3]. It is tightly connected with the Prim’s algorithm of building minimum spanning tree in a weighted graph, and through this is implicitly akin to hierarchical clusterization by the method of single linkage or nearest neighbour [2]. The common feature of these algorithms is the stepwise growing of a spanning tree/cluster/block by adding of closest elements.

iVAT method (“improved VAT”) [3] is a superstructure to VAT and is equivalent in result to application of Floyd–Warshall algorithm in its “identify easiest passes” version to the reordered matrix returned by VAT. iVAT specifically replaces some distances in the matrix by other its distances, thus lessening diversity of distances in the matrix. The iVAT’s idea is simple: if two points as a pair are far from each other but are mediated by a chain of points which all links (distances) are small, one should recognize that the two points are “in fact” close. The effect of iVAT-substitution is that on the heatmap (1) the contrast between between-cluster and within-cluster distances will get stronger, helping to visually uncover clusters-blocks; (2) detectability of clusters of chain-like structure (including strongly elongated, dendric, ring-like) will enhance. iVAT method is performed by the macro when METHOD=VAT /POSTER=YES.

With METHOD=MDS, the macro does reordering of rows/columns of the distance matrix by using multidimensional scaling (SPSS command PROXSCAL) with weighting of distances. It is the macro’s author idea (which doesn’t mean it is a novelty). Each distance *dij*, the matrix element, receives weight (importance) , where *Ri*is the rank of value *dij* in row *i*, *Rj* is the rank of value *dij* in row *j*, *R* is the rank of value *dij* in the triangle of the matrix. PROXSCAL executes the mapping (ordination) in space of dimensionality 1. Rows/columns of the distance matrix are reordered by ascending of coordinates by the dimension. The idea of the method is to force the distances to distribute along the diagonal, giving small distances the priority at that. In VAT method, the size of a distance determines the order of its inclusion in the spanning tree. In MDS method, the size of a distance determines its importance in affecting the ordination. In both methods the final result is that clumps of small distances form blocks threaded on the diagonal.

METHOD=MDS /POSTER=YES applies Floyd–Warshall algorithm in its “identify easiest passes” version to the reordered matrix returned by the MDS method. It has the same effect as iVAT after VAT.

Sources

1. Bezdek, J.C., Hathaway, R.J. VAT: a tool for visual assessment of (cluster) tendency // Proceedings of the 2002 International Joint Conference on Neural Networks. IJCNN'02 – 2002 – Volume 3 – p. 2225–2230. [DOI:10.1109/IJCNN.2002.1007487]
2. Havens, T.C., Bezdek, J.C., Keller, J.M, Popescu, M., Huband, J.M. Is VAT really single linkage in disguise? // Annals of Mathematics and Artificial Intelligence. – 2009 – Vol. 55, article 237 [DOI 10.1007/s10472-009-9157-2]
3. Havens, T.C., Bezdek, J.C. An efficient formulation of the Improved Visual Assessment of Cluster Tendency (iVAT) Algorithm // IEEE Transactions on Knowledge and Data Engineering – 2012 – vol. 24, no. 5, p. 813-822 [DOI: 10.1109/TKDE.2011.33]

**Limitations**

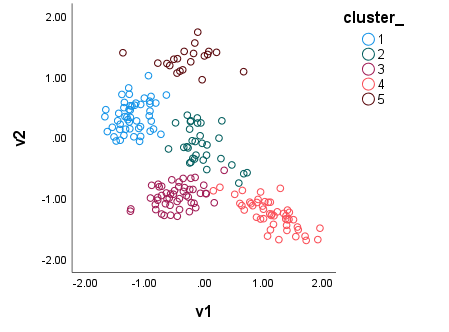
With METHOD=MDS, the matrix compiled for the analysis is up to 700 rows/columns. With METHOD=VAT, the macro does not put limitations, however the recommended matrix size is not above 1000, otherwise heatmap will take long to draw. You can always do the analysis on a random subsample of cases if your data are big.

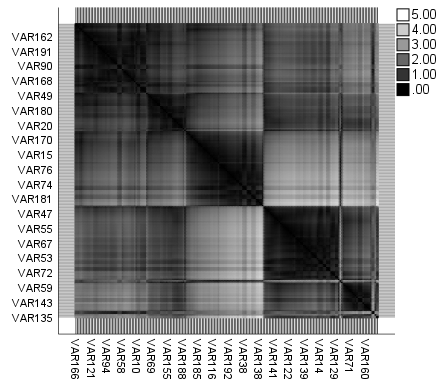
EXAMPLE 1.

proximities v1 v2 /view= case /measure= seuclid /matrix= out(\*) /print= none.

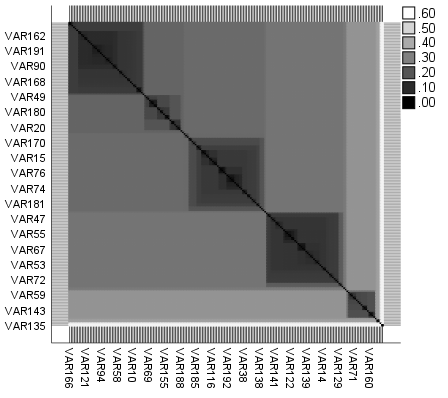
dataset name dist.

!KO\_blockdiag matrix= VAR1 to VAR199.





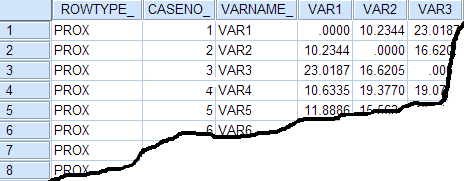
!KO\_blockdiag matrix= VAR1 to VAR199 /poster= YES.



* PROXIMITIES computes squared euclidean distances between cases of the data and saves the matrix as a new dataset which is called *DIST*.
* The macro performs block-diagonalization by VAT method and builds the heatmap. Five clusters can be discerned on it.
* In the second run, the option of “posterization” is added. The five clusters manifest more clearly.

***Matrix structure***

The dataset must be a matrix of pairwise distances (dissimilarities, not similarities). Variable names – matrix columns – up to 8 bytes. Required is the presence of variable VARNAME\_ naming rows in correspondence to columns. Names, which are the values of the variable, should be written in the same case of letters as the identical to them names of the columns. The macro does not require that the rows and the columns go in the same order or their number and list be fully identical: the macro will by itself select from the input matrix rows and columns with the same names and will co-order them so that the compiled matrix be square and diagonalized structure. Variable ROWTYPE\_ and other auxiliary are not necessary in the input matrix.



***Subcommands***

**MATRIX**

Specify variables of the working dataset which are proper the columns of the matrix of distances (dissimilarities). You may list all or just needed columns and in arbitrary sequence. May use “to” to specify by range. If you have similarities, transform them first into dissimilarities the way you find suitable.

*Specification of open range with the help of “?”*. In some instances there may arise need to specify a range between a pair of variables which themselves are not included in the range. Use “?” at the sides, for that. For example, *?VARNAME\_ to ENDVAR?* means all variables found in the dataset between variables *VARNAME\_* and *ENDVAR*, excluding these two. To specify a range which is open from one side, use “?” only from that side. For example: *?VARNAME\_ to VAR100* or *VAR1 to ENDVAR?*.

Since your data are dissimilarities, “diagonal” values – i.e., data in cells on the intersection of rows and columns of the same name – must be zeros, and all other (“offdiagonal”) values must be nonnegative; greater value corresponds to greater dissimilarity.

EXAMPLE 2.

temporary.

sample 0.2.

!KO\_blockdiag matrix= VAR1 to VAR100 /method= MDS /poster= YES .

* SAMPLE command temporarily (under TEMPORARY) selects randomly 20% of rows of the distance matrix.
* The macro takes columns *VAR1* to *VAR100*. The matrix compiled by the macro for the analysis consists of rows and columns that are the intersection of the two lists – selected rows and selected columns.
* The macro performs block-diagonalization by MDS method and “posterization” before building the heatmap.

**ID**

Optional numeric identifier variable of cases (objects). Variable name up to 8 bytes long. There should be no missing values in the variable.

**METHOD**

Indicate the method of block-diagonalization:

VAT - (default/unspecifying) VAT method.

MDS - multidimensional scaling with weighting of distances is performed by SPSS procedure PROXSCAL. It is available in SPSS Statistics Professional Edition or in Categories option.

Both methods often yield very similar results.

**POSTER**

By default/unspecifying and with POSTER=NO, the macro performs only block-diagonalization, i.e., permutation of rows/columns. With POSTER=YES, it does “posterization” after it. Heatmap at “posterization” looks more contrast, and usually clusters (blocks) are easier to see on it. Besides, there somewhat enhances detectability of clusters of chain-like structure (including strongly elongated, dendric, ring-like). Sometimes, though, “posterization” conceals clusters that are close to each other.

“Posterization” with VAT method is known as iVAT (“improved VAT”). “Posterization” affects only the heatmap, it does not affect the matrix being saved. “Posterization” is not the posterization of the obtained image of the heatmap, it is the contrasting of distances in the matrix on which the heatmap is immediately built.

**PLOT**

Heatmap is drawn in greyscale. With PLOT= GREY (default/unspecifying) the higher the value of the element the brighter it is, and with PLOT=RGREY it is opposite – the darker it is. After the keyword, you may add the second keyword LABEL, to label the cells with elements’ values. PLOT=NONE does not produce the heatmap.

**BOUNDS**

This subcommand is not in effect with PLOT=NONE. It sets bounds for the brightness scale.

AUTO - (also default/unspecification) let SPSS automatically define suitable bounds.

OBS - the bounds exactly match with the observed minimal and maximal values in the matrix.

*min* *max* - specify the bounds manually as two numbers, minimum and maximum. Indicate values by and large comparable with the values of the matrix.

Manual specification of the bounds means that you are fixing the brightness mapping on the picture with respect to the size of the elements in the matrix. It becomes possible to compare different matrices with each other by tone immediately.

**MDS**

This subcommand acts with METHOD=MDS. You may choose to use metric or nonmetric multidimensional scaling with PROXSCAL command. Specify INTERVAL (metric), ORDINAL (nonmetric) or SPLINE (in-between metric and nonmetric). By default/unspecifying, MDS=SPLINE.

***Special regimes***

The macro does not obey weighting (however, it doesn’t take in the procedure cases with missing and nonpositive weights). It is not suited for the split state of the dataset (SPLIT FILE). The macro obeys commands selecting cases (SELECT IF, FILTER, USE), including those standing under TEMPORARY command.

# MACRO !KO\_HOPKINS: HOPKINS STATISTIC

Version 1, Mar 2025. Tested on SPSS Statistics 22, 27, 30.

!KO\_hopkins vars= *height weight blpress* /\*Analyzed variables; may use “to”

/jitter= /\*Optional: do data jittering: interval ALL or

/\*interval name-by-name varlist

/trim= /\*Optional: % to trim of outliers, extremes

/z= /\*Do analysis on standardized variables: NO (default) or YES

/numpc= MAX /\*Number of pr components (up to 10) or proportion or MAX or NOPCA

/unifball= /\*Draw in periphery in the data: NO (default) or YES

/m= 0.1 /\*Number of points to generate or their proportion of N

/power= /\*Power parameter: 0, 1, 2, or 3 (default =2)

/mult= /\*Optional: mult parameter, number >=1

/repeat= 30 /\*Number of times to generate m points and compute H (default =1)

/save= /\*Save the m points and objects (and the data): file or declared dataset

/id= /\*Optional: for SAVE, case identifier.

Minimal specification VARS, NUMPC, M.

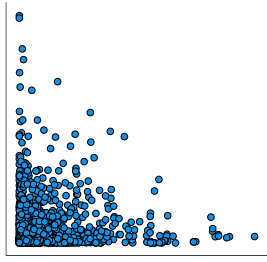
The macro computes Hopkins statistic, which is one of the measures of clustering tendency in the data. It is based on the counterposition of the observed data with the data simulated randomly from uniform distribution. Hopkins statistic allows to judge whether cases of the observed dataset group in clusters or not; consequently, it helps to decide if it is worth doing a cluster analysis of the cases; but it doesn’t tell how many clusters there may be in the data. Input data are quantitative variables. You must have at least 100 cases, but not too many, preferably no more than 2000.

Hopkins statistic (H) varies in the range (0, 1). H on the level of 0.5 tells that there are no clusters (or, rather, that they are on the level of random groupings, possible in a random sample from uniform distribution). Н above 0.7 allows to surmise real clusters, and above 0.8 decisively recommends to do cluster analysis: obvious clusters are present.

The given macro is one of numerous possible implementations of Hopkins statistic, which would differ by how specifically and where the points from uniform distribution are created.

Hopkins statistic H is a bit capricious. It or the present macro make the following assumptions:

* Number of objects (cases) is above 100 but not too many, recommended up to about 2000 for this particular macro. Do random selection before running the macro if you have many more cases.
* In this macro, up to 10 dimensions in the data, and the dimensions are uncorrelated. Use principal components option incorporated in the macro to reduce dimensionality and to decorrelate your data at input. Dimensionality reduction aims not only at speeding up computations but using only important dimensions by which clusters potentially differ.
* This macro supposes the data space is Euclidean and distances therein are Euclidean.
* Continuous scale data. Use jittering option incorporated in the macro in case of discrete/Likert data.
* The macro is not suited for very skewed distributions, such as on **Fig. 1**. Try to transform the data to more symmetric or less skewed form, and also use trimming of extreme tail data by TRIM subcommand.
* Clusters are supposed in this macro to be round or elongated – simple; not curved, star-like or ring-like.
* For this macro, the “no-cluster” null-hypothesis data are random uniform data. Specifically, random uniform round cloud, *p*-ball (if there are *p*>1 dimensions). Use UNIFBALL=YES for your bell-shaped data such as normal. Tip is: try UNIFBALL=NO with your data. If H is high, what evidences of clusters, then try UNIFBALL=YES. Conclude that there are clusters only when both H are high. But if with UNIFBALL=NO H isn’t enough high (doesn’t evidence of clusters), running the macro in the UNIFBALL=YES mode is superfluous.



**Fig. 1**. Strongly skewed data don’t fit with this macro.

EXAMPLE 1.

temporary.

sample 1000 from 5420.

!KO\_hopkins vars= v1 to v8 /jitter= 1 ALL /numpc= .2 /m= .08.

!KO\_hopkins vars= v1 to v8 /jitter= 1 ALL /numpc= 3 /m= .08 /repeat= 10.

!KO\_hopkins vars= v1 to v8 /jitter= 1 ALL /numpc= 3 /m= .08 /repeat= 10 /unifball= YES.

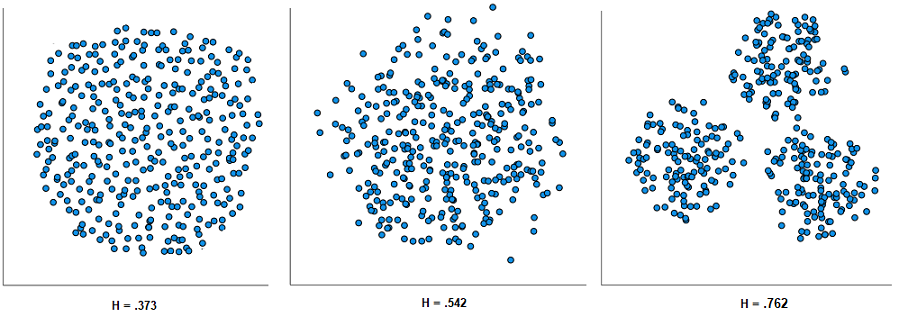
* There are 5420 cases in the dataset. It is more than is recommended for the macro. So the researcher temporarily selects randomly 1000 cases.
* Data in all *V1-V8* are scores on Likert scale 1 2 3 4 5. It is discrete, not continuous data, so the researcher ordered jittering: JITTER= 1 ALL. M=.08 means that generated will be 80 random points (8% of 1000).
* In the first run, there extracted 4 principal components, because the 4th explains yet more variance than 1/5 of the variance of the 1st component, while the 5th explains less than that. Having obtained results in Ouput, the researcher noticed that the first three components are comparable in size, but the 4th is already much weaker. So she chose to replay and ordered 3 principal components in the second run (if there are clusters in the data, they may be responsible for the formation of those strong components[[1]](#footnote-1)). She received 10 values of Hopkins statistic with the mean 0.801. This tells of the presence of clusters (multimodality), but it could also be the consequence of tails of a unimodal distribution.
* To check the latter, the researcher ran the macro with UNIFBALL=YES. This option draws in tails of the distribution of the data, depriving the data cloud its peripheral limb of sparcity. The came out mean value of Hopkins statistic, 0.774, is also high. The researcher concluded that there are clusters in the data worth doing cluster analysis.

**Algorithm**

Hopkins statistic

Let us take interest if there are clusters in a set of *n* *objects* (dataset’s cases) in the *p*-dimensional space of continuous variables. We generate *m* random *points* out of uniform distribution. The region (window) of generation must coincide with the data region (i.e., we “sprinkle” points on the data cloud). We compute distance *ui* from each point *i* to its nearest neighbour object. Randomly select *m* objects out of *n*. Compute distance *wi* from each selected object *i* to its nearest neighbour object (of *n*-1 objects). Hopkins statistic is:

When in the data (i.e., among objects) there are no clusters, and will be close by size, because the random points and the objects will be scattered across the space similarly, random uniformly. So H will be near 0.5. If there are clusters in the data, will be small compared to , because neighbourhood between the objects will be in general tighter than neighbourhood between the points and the objects. H will gravitate towars 1. Possible also, that H aims to 0 – this takes place when the data are regular, they present equal spaces between individual objects. But we are interested primarily in the clustering case. Н above 0.7 allows to surmise clusters, and above 0.8 decisively recommends to do cluster analysis: obvious clusters are present (**Fig. 2**).

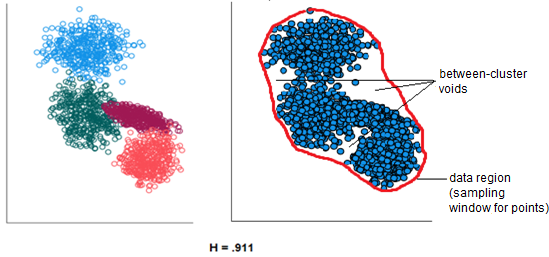


**Fig. 2.** Centre: random no-cluster data; left: data more regular than random; right: cluster data.

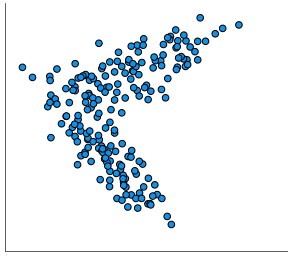
After the null hypothesis that objects distribute random-uniformly, H follows beta distribution with two equal *shape* parameters. If we know *shape* parameter, we can find out statistical significance of the computed H.

*m* should be << *n* (but no less than 10). Recommended is *m* ≈ 0.1*n* or ≈ 0.05*n*. The thing is that beta distribution has its right if each point has its own neighbour object, therefore points must be much less than objects.

The main theoretical and practical difficulty – to decide what is “data region” and where it ends. In other words, the problem is in the sampling window to generate points. If we “spill” points outside the limits of the data region, we enhance and, consequently, H, albeit there may be no clusters. Important is not only the size, but the shape of the sampling window. !KO\_HOPKINS attempts to delineate sampling window by the shape and the size of the data cloud (**Fig. 3**) after giving equal variance to the dimensions.



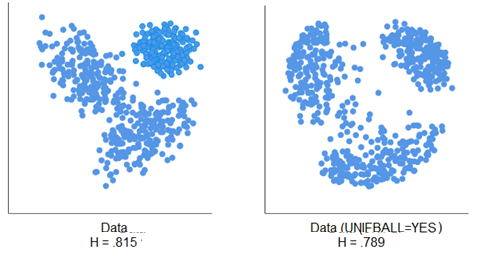
**Рис. 3**. For Hopkins statistic, the sign of clustering is the existence of un-densities between parts of the data cloud. Hopkins statistic tells nothing on the number of clusters, but only on their presence. Hopkins statistic (as realized in !KO\_HOPKINS) is suitable for clusters round or elongated, and not for clusters of complex, curved or ring shape (**Fig. 3a**).



**Fig. 3a**. From the point of view of Hopkins statistic in this macro implementation, here are sooner clusters (they are elongated and contact in one place), rather than no clusters (one curved cloud).

Macro’s operations

1. Input – quantitative data, *n* cases (objects), *p* variables. *n* must be no less than 100 and desirably not more than 2 thousand.
2. Preprocessing.
   1. Data jittering. User-requested if values in the variables are discrete (for example – Likert rating scale). The goal is to turn discrete data into continuous.
   2. Varibles are centered (locus 0 is set to data centroid).
   3. Trimming of outliers. Optional. The requested percent of objects with the greatest Mahalanobis distances to centroid are excluded from the data. Now *n* is the objects left.
   4. z-standardization of variables. Optional. Needed if the variables are in different units of measurement. z-standardization is automatically done if the variable is single or if principal component analysis is cancelled.
   5. Principal component analysis (PCA). Recommended. Done if *p*>1. The goal of PCA is: (i) dimensionality reduction, (ii) decorrelation, (iii) impart equal importance to the dimensions[[2]](#footnote-2). The variables are replaced by orthogonal principal components, which are standardized (their means 0 and variances 1). From now on *p*, the number of dimensions, is the number of principal components (which may be extracted less than there are variables), and data values are the component scores. The number of dimensions may be maximum 10.
   6. Transformation “normal data into uniform” (UNIFBALL=YES). Optional. This operation draws in the periphery of the cloud towards the centre. If the data are normal cloud, the result of the retraction is uniform cloud, with *p*>1 that is the uniform *p*-ball. UNIFBALL=YES is needed for an extra check of clustering tendency (**Fig. 4**).
3. Generation and selection of random points from uniform distribution.
   1. Centre (locus 0) is placed halfway between the centroid of the data and the midrange of the data.
   2. Generation on the data (objects) region and nearby it of random points from box (rectangular) uniform distribution.
   3. Outlining (delineation) of the data cloud by a projection-rejection method: selection of points so that their cloud clothe the cloud of objects.
4. Computation of Hopkins statistic.
   1. Random taking of *m* points (out of selected points) and *m* objects (*m* is usually 0.1*n* or less).
   2. Computation of Euclidean distances to nearest neighbours: from the taken points to objects and from the taken objects to objects.
   3. Computation of H by the formula.



**Рис. 4**. Native cloud with clusters and its distortion under UNIFBALL=YES.

Null cloud

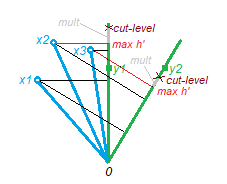
Null cloud, no-cluster cloud, is a population of objects without clusters, but, on the other hand, without regular structure of objects (like grid or web with solitary objects in the nodes). Hopkins statistic takes for null cloud a random cloud from uniform distribution. Macro !KO\_HOPKINS (because it uses outlining dictated form the centre, going in a circle), specifies, that under dimensionality *p* 2 or more that is the uniform *p*-ball (not *p*-cube). H statistic is calibrated such in the macro, that it is expected 0.5 on that null cloud.

Generation of random points (2.2)

The points are produced from rectangular uniform distribution RV.Uniform(min·*mult*, max·*mult*), where min and max are the observed minimum (a negative number) and maximum (a positive number) in the data, and overmeasure *mult≥*1 is explaned subsequently.

Projection-rejection outlining (2.3)

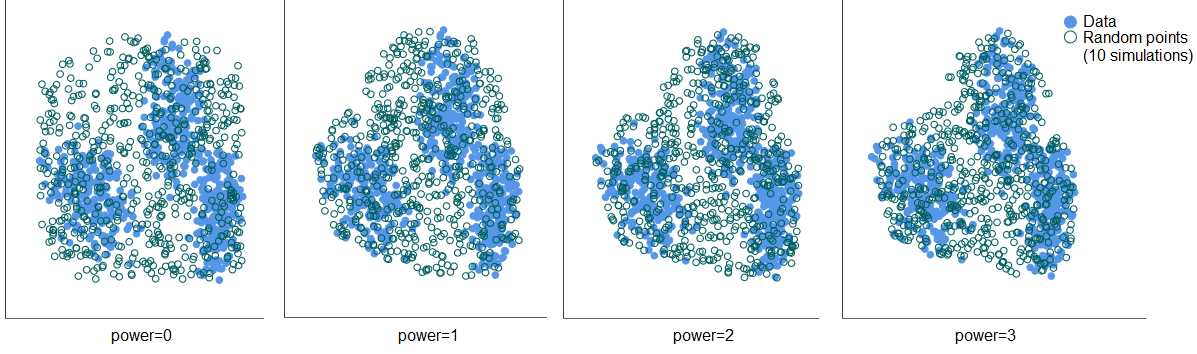
The major challenge in the enterprise of obtaining Hopkins statistic is how to clothe the cloud of observed data with random points. Intuitively, the points should occupy the data region and be limited to it. Else speaking, (1) there where there are objects, points and objects should lie in the environs of each other: mixed, their phases should coincide. (2) Additionally, points should lie in between-cluster space. Outlining by projection-rejection method, accepted in this macro, tries to approximately solve the problem.



**Fig. 5**. The idea of projection-rejection outlining. Three objects *x* and two generated points *y*.

Objects *x* are projected onto the beams of points *y* (**Fig. 5**). The length of projection , where is the beam and the stand-off of *x* from centre 0, and is the cosine of the angle between beam , bearing the object, and beam , bearing the point. cos<0 are taken for 0. Exponent *power* is explained below. Having received projections of all data objects *x*1, *x*2, … on the beam bearing the given point *y*, select the maximal projection of them . And the cut-level is this projection multiplied by coefficient *mult≥*1 explained in further. If , the stand-off of point *y* from the centre, is less than , the point is accepted as lying in the data region, otherwise it is rejected as departuring beyond the data region.

On **Fig. 5**, *power*=1, so the projections are sheer. *power*=2 makes projections slant, canted inward (to locus 0), decreasing them, and the decreese will be the stronger the lesser is the cosine, i.e., the greater is the angle between the object beam and the point beam. *power*>1 leads to more tight, stingy outlining (**Fig. 6**), because it takes out of the game in this place of space those objects, angularly far form the place. *power*=0, on contrary, turns projection into rotation of beam into beam , so very far-away objects impact on the fate of the point. We recommend to use *power* 1 or 2. 2 is the default.



**Рис. 6**. Parameter *power* and its impact on the shape of outlining of data by points.

Parameter *mult*

With the increase of *p/n*, outlining by the projection-rejection method leads, unfortunately, to the delimitation of phases of objects and points. The reason is the curse of dimensionality. In random data of high dimensions, objects seek to lie about equal distances from each other, and each object seeks to be external, belong to the convex hull of the data cloud – when the objects are not plenty. It turns out that inside the data cloud there are almost no objects[[3]](#footnote-3). But there points will be, selected by the outlining. Points in conditions of relatively high *p* and not large *n* will tend to be inwards of objects. And points will generally be closer to closest objects than objects to closest objects. It will lead to the decrease of H. (In whole, projection-rejection outlining has the goal to remove points from beyond-cluster space and leave them in between-cluster space. But in high dimensionality, in connection with its sparsity, the difference between between-cluster and beyond-cluster regions quickly vanishes.)

In order to compensate for the noted tendency of phases delimitation one may introduce, at outlining, an *overmeasure* for points: let a point be allowed to lie more peripherally (see **Fig. 5**) than that mark of maximal projection which projection-rejection outlining computes. That overmeasure could be a simple, constant for all points, multiplier *mult*, which magnitude depends on *p* and *n* (and on *power*). The multiplier is selected purely empirically such that in simulations with random data from uniform distribution H would appear with arithmetic mean close to 0.5 (**Tab. 1**). Though there will be no coincidence of phases: points will occupy conversely now somewhat a greater volume than objects, but H will be centered on the size 0.5.

**Tab. 1**. Picked values of *mult*, at which the mean value of H, obtained in simulations of random no-cluster data (uniform *p*-ball, when *p*>1), is close to 0.5.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | ***p*:** | **1** | **2** | **3** | **4** | **5** | **7** | **10** |
| ***n*:** |  |  |  |  |  |  |  |  |
| **3000** |  | **1** |  |  |  |  |  |  |
| **2000** |  | **1** | **1.01/1.015** | **1.03/1.048** | **1.06/1.1** | **1.094/1.165** | **1.173/1.317** |  |
| **1000** |  | **1.003** | **1.02/1.02** | **1.043/1.07** | **1.082/1.133** | **1.122/1.217** | **1.212/1.39** | **1.345/1.645** |
| **700** |  | **1.005** | **1.024/1.03** | **1.055/1.084** | **1.098/1.16** | **1.144/1.249** | **1.238/1.44** | **1.379/1.695** |
| **500** |  | **1.008** | **1.03/1.04** | **1.068/1.105** | **1.114/1.188** | **1.166/1.285** | **1.267/1.485** | **1.413/1.745** |
| **300** |  | **1.013** | **1.04/1.06** | **1.089/1.14** | **1.146/1.24** | **1.203/1.348** | **1.32/1.564** | **1.466/1.822** |
| **200** |  | **1.019** | **1.06/1.08** | **1.116/1.175** | **1.177/1.288** | **1.24/1.41** | **1.367/1.634** | **1.52/1.886** |
| **100** |  | **1.038** | **1.1/1.13** | **1.18/1.258** | **1.25/1.399** | **1.326/1.537** | **1.463/1.77** | **1.624/2.004** |

Above slash – value for *power*=1; under slash – value for *power*=2.

We approximated values in **Tab. 1**, dependent on *n* and *p*, regressionally. We used program DataFit vers. 9, allowing to fit a lot of regressional models, and chose an equation of the form

Y = a+b\*ln(*n*)+c\**p*+d\*ln(*n*)^2+e\**p*^2+f\*ln(*n*)\**p*+g\*ln(*n*)^3+h\**p*^3+i\*ln(*n*)\**p*^2+j\*ln(*n*)^2\**p*

for case *p*>1, and an equation of the form

Y = a+b\*ln(*n*)+c\*ln(*n*)^2+d\*ln(*n*)^3+e\*ln(*n*)^4

for case *p*=1.

The equations themselves turned out to be these. By these equations the macro computes *mult* parameter.

For *p*>1, *power*=1:

*mult* = 1.8001412953476-.37821110173515\*ln(*n*)+.200639641499899\**p*

+5.34921458279123E-02\*ln(*n*)^2-2.29345019432612E-03\**p*^2

-3.37379264269758E-02\*ln(*n*)\**p*-2.28570712583439E-03\*ln(*n*)^3

-2.85869384260207E-04\**p*^3+1.21163113175788E-03\*ln(*n*)\**p*^2+7.63762113783788E-04\*ln(*n*)^2\**p*

(R^2 = 0.9997846286)

For *p*>1, *power*=2:

*mult* = 1.55402545812428-.285142518531347\*ln(*n*)+.316937136127879\**p*

+3.54656315363041E-02\*ln(*n*)^2-4.85182536412298E-03\**p*^2

-4.64194001390029E-02\*ln(*n*)\**p*-8.44286652372263E-04\*ln(*n*)^3

-7.80112278128893E-04\**p*^3+3.03660728431481E-03\*ln(*n*)\**p*^2-2.99246137741637E-04\*ln(*n*)^2\**p*

(R^2 = 0.9997016488)

For *p*=1:

*mult* = 2.48919971705474-.843966420677826\*ln(*n*)+.183812420447206\*ln(*n*)\*\*2-1.80707757504798E-02\*ln(*n*)\*\*3

+6.71576972339337E-04\*ln(*n*)\*\*4

(R^2 = 0.9998991173)

For *power*=0 and 3 the macro does not compute *mult*. The user has to supply *mult*.

As was shown before, parameter *mult*, i.e. overmeasure, is used twice. First, at generating the initial “box” of points from uniform distribution. Second, at selection of points by projection-rejection outlining.

Significance of H

Sig. of the null hypothesis that there are no clusters in data, against one-sided alternative hypothesis that there are clusters in data, = 1-CDF.Beta(H,*shape*,*shape*), where *shape* is the parameter in beta distribution. Theoretically, in conditions of infinite Poisson process in space, *shape* = *m*. In real situation of our macro, where there is a limited in space null cloud in the form of *p*-ball, *shape* only approaches *m* and is dependent on *n* and *p*. In **Tab. 2**, shown are some measured (in SPSS procedure P-P plot) values of *shape* with *m* = 25.

**Tab. 2**. *Shape* parameter with *m* = 25.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | ***p*:** | **1** | **2** | **3** |
| ***n (m/n)*:** |  |  |  |  |
| **1000 (.02)** |  | **25.0** | **23.2** | **21.7** |
| **500 (.04)** |  | **24.4** | **22.9** | **21.2** |
| **200 (.10)** |  | **22.8** | **22.0** | **21.0** |

As one can see, with the increase of *m/n*, as well as the increase of *p*, *shape* falls. It is possible to model the dependence of *shape* on *m*, *n* and *p*, but at present this work has not been done. Therefore the macro does not compute Sig.

Reading on Hopkins statistic

* Hopkins, B., Skellam, J.G. A new method for determining the type of distribution of plant individuals // Annals of Botany. – 1954 - 18(2). - pp. 213-227.
* Adolfsson, A., Ackerman, M., Brownstein, N.C. To cluster, or not to cluster: An analysis of clusterability methods // Pattern Recognition. – 2019. – 88. – p. 13–26.
* Panayircit, E., Dubes, R.C. A test for multidimensional clustering tendency // Pattern Recognition. – 1983. – 16. – p. 433-444.
* Banerjee, A., Dave, R.N. Validating clusters using the Hopkins statistic // FUZZ-IEEE 2004. - 25-29 July, 2004, Budapest, Hungary.
* Cross, G.R., Jain, A.K. Measurement of clustering tendency // Theory and Application of Digital Control. - New Delhi, India. – 1982.

**Performance**

Speed depends primarily on space dimensionality *p* and the number of points being used *m*. In a space of high *p* (for example, 10) points are selected by projection-rejection method very slowly, and accumulation of *m* points, especially when *m* is large, takes much time. Also, speed depends on the number of objects *n*.

***Subcommands***

**VARS**

Analyzed variables. Full list and/or via “to”. It must be quantitative features. If there are missing values in the data, the macro will exclude them listwise, i.e., the complete case will be excluded from the analysis if it is missing at least in one variable.

**JITTER**

Preprocessing 1. Jittering of objects. Use if your data are discrete, for example they are rating scores on a Likert type scale. Data jittering turns discrete values into continuous. Continuety of variables is a necessary condition for Hopkins statistic.

Indicate the number – the interval between levels of the scale, and after it name-by-name list of discrete variables of VARS, or ALL (what means “all variables VARS”). For example, JITTER= 1 ALL means that all variables VARS are discrete, with unit interval between adjacent levels of their scale. JITTER= 1 VAR1 VAR2 VAR5 means that of VARS, discrete are only variables VAR1, VAR2, VAR5 (and in all of them the interval between adjacent levels of the scale equals 1), while other variables VARS are continuous in their values.

Important:

1. interval may be specified just one. If scales in your different variables are of different interval width (for example, there’s scale 1 2 3 4 and there’s scale 10 20 30), transform the variables first, making the interval the same width everywhere.
2. variable names specified in s/c JITTER must be up to 8 bytes long and written in the same letter case as they are written in the dataset.

**TRIM**

Preprocessing 2. Trimming of specified percent of outliers/extremes. Outliers tend to increase Hopkins statistic. Indicate percent of objects, for example TRIM=2.5. Then 2.5% of objects (cases) with greatest Mahalanobis distances from data centroid will be excluded from the analysis. Percent must be greater than 0 and not greater than 10.

**Z**

Preprocessing 3. Z=YES standardizes variables before principal component analysis. Standardization is needed if measurement units are different in the variables. If principal component analysis is cancelled (NUMPC=NOPCA), variable standardization is done even if it is not requested (Z=NO, the default).

**NUMPC**

Preprocessing 4. The subcommand is required if VARS is more than one variable. Principal component analysis (PCA) replaces input variables by standardized principal components. The goal is to reduce dimensionality (the macro permits maximum 10 dimensions in data) and to decorrelate dimensions (what is important for the procedure of outlining data – see “Algorithm”). From now on “data” – it will be the principal components.

Specify one of the following:

*number of components* - integer (not greater than 10), how many principal components to extract.

*proportion* - decimal proportion >0 and <1. Extract all principal components with eigenvalues >*proportion*·*λ1*, where *λ1* is the eigenvalue of the 1st component.

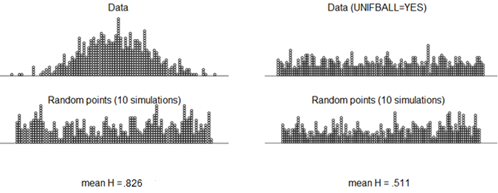
MAX - extract all nonzero principal components (but not greater than 10).

NOPCA - don’t do principal component analysis. This option supposes that variables VARS are up to 10, and that they don’t or almost don’t correlate.

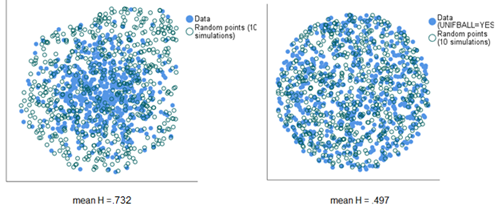
Having done PCA, the macro displays eigenvalues and the percent of variance explained, so you can decide how may components to extract (retain) at the next run of the macro. One would extract only strong components. For instance, NUMPC=0.25 is not a bad choice, it retains only components which standard deviation is greater than half of the 1st component’s standard deviation. All the extracted components are considered by the macro to be equal importance dimensions – standardizing their scores.

**UNIFBALL**

Preprocessing 5. By default and with UNIFBALL=NO, the macro analyzes the undistorted data cloud. With UNIFBALL=YES, it analyzes the data cloud with retracted periphery (tails). The farther the object stood apart from the centroid, the more way it displaces towards the centroid. If the data cloud was normal distribution without clusters, UNIFBALL=YES turns it into uniform distribution (**Fig. 7**); in case of *p*-dimensional normal distribution if will be uniform *p*-ball (**Fig. 8**). See also **Fig. 4**.



**Рис. 7**. Normal distribution of data Data (left) and it turned into uniform (right).



**Рис. 8**. Normal 2D distribution of data Data (left) and it turned into 2D uniform ball (right).

Hopkins statistic, as implemented in the macro, is based on the null hypothesis that the data come from uniform distribution, globular at *p*>1. For the macro, data from normal and, wider, from bell-shape unimodal distribution are not cluster-free data, rather, they are single-cluster data: Hopkins statistic will be increased. Therefore, if in the UNIFBALL=NO regime you see high Hopkins statistic, do not hurry to conclude that there are *clusters* in the data. Run the macro the second time, now in the UNIFBALL=YES regime. If in this mode Hopkins statistic is high too, then conclude the data consist of clusters.

Hopkins statistic under UNIFBALL=YES is often lower than under UNIFBALL=NO. If under UNIFBALL=YES it exceeds 0.6 (while under UNIFBALL=NO it is above 0.7), you may suspect the presence of not clearly expressed clusters.

**M**

Number of random points to generate/select in the data region. Specify integer not less than 10. Or specify decimal proportion not greater than 0.1. This proportion is the fraction of *n*, the number of objects (cases) in the data. Usually the proportion is from 0.05 to 0.1.

**POWER**

Power in which to take cosine at projection-rejection selection of points (see “Algorithm”). Specify 0, 1, 2, or 3. By default, POWER=2. POWER 0 or 3 we don’t recommend. POWER is ignored if VARS is a single variable or single component is extracted.

**MULT**

Optional subcommand with which you can specify your own value for parameter *mult* (see “Algorithm”). By default, *mult* is computed by the macro with a formula, except when POWER= 0 or 3.

**REPEAT**

How many times to generate/select points and to compute Hopkins statistic. Indicate positive integer. By default, REPEAT=1.

**SAVE**

This subcommand is for saving data after their preprocessing (whichever was ordered), saving selected *m* points and *m* objects used to compute Hopkins statistic. Specify .SAV file to save or the name of declared dataset.

In the file/dataset there will be variable *SOURCE\_* with values 0, positive and negative integers. 0 flags *n* data cases, i.e., all the objects. 1 flags *m* generated and selected random points for the 1st one of REPEAT times computed Hopkins statistic, and -1 flags *m* randomly selected objects to compute it. 2 and -2, respectively, flag points and objects on which the 2nd Hopkins statistic was computed. And so on, up to figure REPEAT.

EXAMPLE 2. To see data cloud and generated points.

dataset declare savedata.

!KO\_hopkins vars= v1 to v8 /z= YES /numpc= 2 /m= .08 /repeat= 10 /save= savedata.

dataset name h.

dataset activate savedata.

select if source\_>=0.

recode source\_ (0=0) (else=1).

value labels source\_ 0 'data ' 1 'points '.

graph /scatterplot(bivar)= pc1 with pc2 by source\_.

* Declared is dataset *SAVEDATA*. The macro saves materials used to compute 10 pieces of Hopkins statistic to this dataset.
* Dataset with the Hopkins statistic receives name *H*.
* Dataset *SAVEDATA* gets activated. SELECT IF removes from it the objects selected for computations, leaving only the data themselves (all objects) and generated/selected points. RECODE recodes all numbers 1, 2, …, 10 (10 pieces of statistic was computed) into 1.
* Scatterplot is built.
* Data are the data after preprocessing. In this case, preprocessing consisted of standardization of the input variables and extraction of two principal components, their scores are standardized.

**ID**

You may indicate numeric case identifier variable in your input dataset. Identifier makes sense when you specified SAVE. If the data had missings or if you used s/c TRIM, the data saved by s/c SAVE will have less cases (*SOURCE\_*=0) than there was in the input dataset. The identifier will help trace cases.

***Special regimes***

The macro does not obey weighting (however, it doesn’t take in the procedure cases with missing and nonpositive weights). It is not suited for the split state of the dataset (SPLIT FILE). The macro obeys commands selecting cases (SELECT IF, FILTER, USE), including those standing under TEMPORARY command.

1. Moreover, one may surmise that there are 3+1, i.e. 4 clusters. [↑](#footnote-ref-1)
2. We are aware that equalizing variances of the dimensions is some distortion of the data. Clusters and between-cluster voids may change their shape. But the contribution of dimensions needs be equalized, because the formula of Hopkins statistic allots dimensions equal contribution; besides, projection-rejection outlining used by the macro implies about equal variances of dimensions. [↑](#footnote-ref-2)
3. In the limiting case, where *p*=*n*-1, we practically always get simplex, a figure with no objects “inside”. [↑](#footnote-ref-3)