

## **Clustering words to match conditions: An algorithm for stimuli selection in factorial designs**

Marc Guasch<sup>\*1</sup>, Juan Haro<sup>1</sup> and Roger Boada<sup>1</sup>

<sup>1</sup>*Universitat Rovira i Virgili. Tarragona. Spain*

With the increasing refinement of language processing models and the new discoveries about which variables can modulate these processes, stimuli selection for experiments with a factorial design is becoming a tough task. Selecting sets of words that differ in one variable, while matching these same words into dozens of other confounding variables is time consuming and error prone. To assist experimenters in this thankless task, we present a simple method to perform it with little effort. The method is based on K-means clustering as a way to detect small and tight clusters of words that match in the desired variables. We have formalized the procedure into an algorithmic format, that is, a series of easy-to-follow steps. In addition, we also provide an SPSS syntax that helps in choosing the correct size of the clustering. After reviewing the theory, we present a worked example that will guide the reader through the complete procedure. The dataset of the worked example is available as a supplementary material to this paper.

Factorial designs are one of the most fertile methods of study in psycholinguistics, (but see Baayen, 2004, 2010, and Cohen, 1983, for critical assessments). A factorial design in this field often involves the discretization of a continuous variable (e.g., word frequency, mean value of concreteness rated in a Likert scale, etc.), and the experimental control of all those other variables that may covariate with the one of interest. This experimental control in factorial designs has become harder over the years. Some decades ago, Cutler (1981) already noted the increasing difficulty in selecting controlled materials for psycholinguistic studies due to the

---

\* **Acknowledgements:** This research was funded by the Spanish Ministry of Economy and Competitiveness (PSI2015-63525-P) and by the Research Promotion Program of the Universitat Rovira i Virgili (2014PFR-URV-B2-37). **Corresponding author:** Marc Guasch. Department of Psychology and CRAMC. University Rovira i Virgili. Crta. de Valls s/n., 43007-Tarragona. Spain. Telephone number: +34 977 55 85 67. E-mail: marc.guasch@urv.cat

“continual discovery of new confounds” (p. 65). One of the examples used in her paper was about word frequency, where the author complained about having to match items not only on surface frequency (i.e., the frequency of a particular word form), but also on combined frequency (i.e., the frequency of all the inflected forms of a word). Day by day, confounds concerning word frequency have increased dramatically. Now there is a whole family of frequency related variables that could be controlled in a word recognition experiment, including but not limited to subjective familiarity, age of acquisition, contextual diversity, or bigram and trigram frequency.

The problem when selecting materials is two-fold: on the one hand, it is necessary to have databases of many psycholinguistic variables for a large amount of words. Fortunately, psycholinguists have developed several tools to have convenient access to different word properties (i.e., sublexical and lexical variables, subjective data, etc.). Some notable tools of this type in Spanish, in the form of web sites, are EsPal (Duchon, Perea, Sebastián-Gallés, Martí, & Carreiras, 2013), NIM (Guasch, Boada, Ferré, & Sánchez-Casas, 2013), or NIPE (Díez, Fernández, & Alonso, 2006). Similar tools in other languages such as English also exist, like the MRC Psycholinguistic Database (Coltheart, 1981), or the N-Watch software (Davis, 2005) available also in other languages (e.g., Spanish: Davis & Perea, 2005). By combining these tools with other published databases, researchers can start with little effort with a good pool of items. Then, they have to face the second problem: to select just those words that fulfill an extensive list of experimental requirements.

To overcome this part of the problem already predicted by Cutler (1981) that “psycholinguists will literally be lost for words” (p. 69), some automated procedures have been developed. Two remarkable examples of computerized solutions are Match (Van Casteren & Davis, 2007), and Stochastic Optimization of Stimuli (SOS; Armstrong, Watson, & Plaut, 2012). Match automates the process of picking items for experimental sets by selecting the best-matching pairwise items from different lists of candidates. The matching criterion is based on the normalized standard Euclidean distance between item pairs taking into account the values of all the variables to be controlled, and trying to minimize the sum of the squares of all the distances between lists. To reach this goal, the program operates in a recursive manner creating tuples of matched items from each list of candidates and, when a solution is found, backtracking to try to find a better solution.

With the same aim as Match, SOS allows for both pairwise and groupwise matching solutions. Additionally, the user can establish a

powerful set of constraints that the resulting solution should meet, either as hard constraints (i.e., all-or-none rules that the values of a variable have to fulfill), or as soft constraints (i.e., desired relationships between variables). SOS also allows an evaluation of how well the constraints have been satisfied, and to assess the degree of representativeness of the final item selection.

The aim of the present work is to present a new method for helping in the selection of controlled materials for psycholinguistic studies with a factorial design. This method is halfway between a full manual selection, and a completely automated one. Therefore, it contains some of the advantages of both approaches. The problems with manual procedures have been clearly described by Armstrong et al. (2012), (see also Van Casteren & Davis, 2007). Picking items by hand is tedious, time consuming, error prone, and it is difficult to evaluate the representativeness of a set of materials chosen this way. By the other hand, a fully automated selection sets aside the experimenter, and leaves open the question if a better solution could have been found.

Our proposal is concise enough to overcome the problems of the first approach, while at the same time it is flexible and does not output one closed solution. Instead, it offers a tentative solution that the experimenters can debug and refine according to the experimental requirements. Another appeal of our proposal is the simplicity of all the procedure, as any researcher with a statistical package such as SPSS can obtain a valid solution in a few minutes.

The core of our method relies on the cluster analysis technique. Cluster analysis is a general name that encompasses several different algorithms to classify a sample of elements into different groups, according to a set of variables. When performing this grouping, the goal is to minimize the differences between the members of the same group on the measured variables, while maximizing the differences between groups. Cluster analysis methods are exploratory in nature, and their aim is to reveal the natural structure of the data without having a previous theoretical framework. Thus, they are very useful for many fields when studying large amounts of data (e.g., biology, botany, medicine, psychiatry, astronomy, marketing, etc.). For instance, they are used to create meaningful taxonomies of species, to classify patients or diseases, to identify distinct groups of potential customers, etc. In the psychology domain, there is also a long tradition of using clustering methods. They have been used to detect behavior or personality patterns, to classify organizational structures, educational styles, learning methods, etc. For instance, Lorr and Strack

(1994) used a clustering procedure to detect personality profiles of police candidates, whereas Hillhouse and Adler (1997) used it to categorize nursing stress subtypes. Even in the more reduced domain of psycholinguistics, cluster analysis is also a known technique. In this domain it has been used applied to participants, (e.g., as a tool to examine differences in language learners; Sparks, Patton, & Ganschow, 2012), as well as applied to items (e.g., to study how items group in a semantic space; Troche, Crutch, & Reilly, 2014).

However, the use of clustering that we present here is not the common use of the technique (i.e.: for trying to understand the data). Our approach consists in exploiting the group matching capabilities of cluster algorithms to find small clusters of stimuli as similar as possible in a given set of variables. From the variety of available cluster models and clustering procedures (e.g., hierarchical, partitioning, density-based, model-based, etc.; see Rokach & Maimon, 2005, for an overview of the main clustering methods), we have chosen the K-means algorithm, as it is a relatively simple method that at the same time suits our purposes (see Jain, 2010, for a comprehensive review of the K-means algorithm). In the K-means clustering, the goal is to group  $n$  observations into  $k$  clusters. Each cluster has a center computed as the mean of all the instances that belong to it. Then, each observation is assigned to the nearest cluster according to its center. Thus, the algorithm operates in an iterative manner starting from an initial set of cluster centers, assigning each observation to the nearest cluster, and then computing the new cluster centers. This procedure is repeated until a stopping criterion is reached.

In the next section, we will formalize this idea of using K-means clustering to select well-controlled experimental stimuli, by detailing a series of steps that can be easily followed using the SPSS software package. Of note, unlike other methods of clustering, in order to perform a K-means clustering it is necessary to know beforehand the optimal value of  $k$ , and this choice will sharply determine the quality of the final selection. Thus, special attention will be paid to the optimal way for deciding the initial number of clusters. Then, we will present a worked example with a dataset available as a supplementary material.

### **The algorithm**

The general idea of using K-means clustering to select matched stimuli for factorial designs in psycholinguistic studies is simple. By dividing a large set of items into  $k$  groups of  $n$  observations each, the members within each group will be as much equal between them as

possible. Ideally, if each of these tight groups contains just one member belonging to each of the experimental conditions, the result will be a pairwise matched set of materials. Unfortunately, this ideal situation is hardly achieved, but our method aims to get as close as possible. In what follows, we detail the procedure and its steps.

**Preparing the data.** The starting point is a large set of words and their corresponding values in the variables to be controlled. Additionally, it is necessary to create a new variable (e.g., “Condition”) to classify the items into the desired experimental conditions (i.e., it is necessary to discretize items through the critical variables). Of note, the variable “Condition” can comprise a large number of experimental levels, given that the number of experimental conditions does not hamper the application of the procedure. On the other hand, the procedure also applies to experimental designs with more than one independent variable. In such cases, the researcher has to combine those variables in order to create the experimental conditions. For instance, suppose that we have a 2x2 factorial design: word length (short and long words), and word frequency (low and high frequency words). This leads to four experimental conditions: short/long words, with low/high frequency. Thus, in this case, the variable “Condition” should comprise four levels.

Then, all variables must be converted to z-scores. Normalization is a necessary step because K-means clustering is strongly influenced by the magnitude of the variables. Without normalization, the partitioning of the dataset is driven mainly by the few variables with the highest magnitudes, relegating the others to a secondary role. The conversion of all variables to z-scores overcomes this problem. Other computerized solutions such as Match (Van Casteren, 2007) and SOS (Armstrong et al., 2012) are also sensitive to the convenience to normalize the initial raw data.

**Selecting the appropriate number of clusters.** Once all variables have been converted to z-scores we can compute the K-means clustering. We aim to partition the initial set of words into as many groups as possible (the more clusters, the tightest the groups). However, we have to satisfy the restriction of obtaining enough clusters with at least one member of each experimental condition, as to reach the desired number of items per condition. Thus, according to this restriction some of the obtained solutions will be useful, while others will be useless.

As stated before, this kind of clustering requires knowing in advance the number of partitions needed (i.e., the value of  $k$ ), and the success of the

process depends mainly on the selection of this value. We cannot know certainly the optimal value of  $k$ , but we can compute some indices that can help us to approximate it. First, we can examine the minimum number of clusters (*k-min*) that can lead to a valid solution. This is quite simple, as *k-min* value is always two. The resulting division in two parts would probably generate two clusters containing items of all the experimental conditions, thus, assuring enough items to choose from. However, the cohesion of the items within those clusters would be very low, which does not ensure a good final matching between conditions.

The opposite extreme is found when we establish the maximum number of clusters (*k-max*). This value corresponds to the number of observations in the condition that contains fewer stimuli (note that it is not a requirement to have the same number of items per condition in the initial word set). It is reasonable to think that if, for example, we have only ten items belonging to a given condition in our starting set of materials, we cannot ask for eleven partitions of it. If we did it, it is sure that at least one cluster would not contain an item belonging to that condition, resulting in discarding the rest of items in that cluster. However, to choose a *k-max* of ten does not ensure that each cluster will have at least one item from each condition. Therefore, in any resulting solution there will be a certain amount of valid clusters (i.e., clusters that contain at least one item belonging to each experimental condition), and a certain amount of invalid clusters (i.e., clusters that do not allow researchers to select at least one item for each experimental condition). Our goal then is to select a value of  $k$  between *k-min* and *k-max* that provides a number of valid clusters enough to contain at least the desired number of stimuli per condition. Additionally, to approach to an optimal solution we must look also for valid clusters being the distance of each item and the center of its cluster as small as possible.

For exploratory purposes, a rule of thumb for selecting a  $k$  value can be to choose the middle point within the range of possible values. However, to get closer to the optimal  $k$  value some additional steps can be followed. Our approach here is to study the results of multiple runs of K-means clustering checking all the possible  $k$  values, and to choose a suitable one according to two indices. As running multiple analyses involves non-trivial computations, we will present in the worked example (see below) an SPSS Syntax that can do the job in a few minutes.

The first index that we can use to approach to an optimal value of  $k$  is the number of valid items across the possible  $k$  values. In any given grouping, we can only use those items that have a counterpart from the rest of conditions in the same cluster. For instance, in an experiment with two

conditions, if we have a cluster with one item from condition one, and two items from condition two, we can only use one item from condition two. Thus, it is possible to compute the number of usable items for each value of  $k$ . For the final analysis, we will have to select a  $k$  value that provides at least the desired number of items per condition.

The second index that we can consider is the sum of the distances from each item to the center of its cluster, for each  $k$  value. If the number of clusters is too low, there will be a large amount of valid but heterogeneous clusters. Items in those clusters will be far from the center of their respective clusters, and the total sum will be large. In contrast, the higher the  $k$  value, the lower the sum of distances and the better the match within the items of a cluster. Graphically it can be represented as a scree plot similar to that used in a principal components analysis or a factor analysis. Usually a scree plot shows a descending curve with a bend after which the curve turns into a flat line. In the aforementioned analyses, the optimal point is that corresponding to the bend, since going further to that point causes the problem of overfitting the model. However, our goal is not to create a valid model of the data but to look for the best match among a group of items. Being so, we do not need to worry about overfitting and usually the higher values of  $k$  will be the better.

In sum, for selecting the appropriate value of  $k$  we can explore the number of valid items, and the sums of distances. Any value that gives as a result the required number of valid items, while minimizing the sum of distances, is a potential value to obtain a useful solution. All these measures can be used to determine beforehand if our corpus for selecting materials is suitable for our needs (i.e.: if it contains enough matching items). Note also that the use of these indices does not lead to the optimal solution, but to one of the potentially suitable solutions.

**Selecting the items.** Once  $k$  has been estimated, the next step is to carry out the K-means clustering asking SPSS to create two new variables: one for the cluster membership of each item, and another to keep a record of the distance of each item to its cluster center. From here, the experimenter has to follow some simple additional steps. The first one would be to remove from the corpus those items belonging to invalid clusters (i.e.: groupings with missing items in one or more conditions). Then, it is possible that a valid cluster has not the same number of items per condition. Suppose, for instance, that in an experiment with two conditions there is a cluster with two items from condition one, and one item from condition two. In this case, one item from condition one should be removed. As a

criterion for selecting between the two possible items from condition one, the experimenter can choose among several strategies. In order to avoid biases in the item selection, the experimenter should keep constant the selection strategy. One possible strategy would be to systematically remove the item whose distance from the center of the cluster is more different than the distance from the cluster center of the item in condition two. Another strategy would be to select the item with the lowest distance to the cluster center.

After this item trimming process, we will obtain either the exact number of desired items per condition, or a greater number. In this last case, the experimenter should remove one item per condition from the same cluster until the desired number of stimuli is reached. Again, the criterion for selection should be kept constant to avoid biases (e.g., removing those items with the higher distances to the center of their respective clusters).

**Assessing the solution.** The last step is to check for the validity of the resulting stimuli set. The most direct way to do it is to perform a One-Way ANOVA setting as a factor the variable “Condition” and as dependent measures all the list of controlled variables before standardization. If all  $p$  values are above the selected significance level, the procedure ends. If not, we can go back to the surplus-removing steps and to choose another set of items. Another possibility would be to repeat the selection process starting from another of the valid values of  $k$ . To summarize, Figure 1 describes the whole procedure in an algorithmic format.

### Worked example

To exemplify the use of the algorithm described above we devised an example. Suppose that we have a starting point of 200 words with their respective measures on seven different variables. From this pool, we want to obtain three experimental conditions of 30 items each, where the three conditions are different in a critical variable, but they are matched in the other six variables.

The dataset used to go through this example can be found as supplementary material to this paper. To construct our starting dataset we selected a list of 200 English words ranging from three to ten letters. Then, these words were introduced into N-Watch (Davis, 2005) to obtain measures of frequency (in number of occurrences per million), word length, number of syllables, number of neighbors, number of higher frequency neighbors, mean bigram frequency, and imageability ratings. The selection of these variables was not guided by theoretical issues, but by the aim of



better testing our matching procedure. Indeed, the selected variables had different scales, different distributions, and some of them were correlated among them (e.g., word length and number of syllables). Thus, a set like this constitutes a realistic example of items selection. Next, we began with the application of the algorithm.

- 
- Step 1.** Discretize critical variables to create conditions.
- Step 2.** Normalize all the variables to be controlled.
- Step 3.** Determine the number of clusters for K-means clustering ( $k$ ):
- Step 3.1.** Define  $k$ -max.
- Step 3.2.** Examine the number of valid items across  $k$  values.
- Step 3.3.** Examine the sums of distances across  $k$  values.
- Step 4.** Run the K-means clustering with the selected  $k$  value.
- Step 5.** Select final items from the raw solution:
- Step 5.1.** Delete invalid clusters.
- Step 5.2.** Delete the non-matched items in valid clusters.
- If the number of valid items is greater than the intended number of items per condition:
- Step 5.3.** Delete one item of each condition on valid clusters.
- Step 6.** Assess the resulting set.
- If there are significant differences in any of the variables to be controlled:
- Change the item removing criterion, and return to **Step 5.3.**, or
- Change the item removing criterion, and return to **Step 5.2.**, or
- Return to **Step 3** and choose another suitable  $k$  value.
- 

**Figure 1. Algorithm for the selection of experimental materials matched in a set of variables across  $n$  conditions, using K-means clustering.**

**Step 1** was to discretize the critical variables to create the experimental conditions. We choose word frequency as the critical variable, and the stimuli set was divided into three conditions based on that variable. Words were classified (by creating a new variable called “COND”) as low-

frequency words if their frequency were less than 15 occurrences per million (Condition 1;  $M = 7.74$ ,  $SD = 3.82$ ), as medium-frequency words if their frequency were equal or greater than 15 and less than 60 occurrences per million (Condition 2;  $M = 34.31$ ,  $SD = 13.31$ ), and as high-frequency words if their frequency were greater than 60 occurrences per million (Condition 3;  $M = 176.91$ ,  $SD = 169.17$ ). This procedure resulted in 70 low-frequency words, 60 medium-frequency words, and 70 high-frequency words. The properties of the selected words by condition are shown in Table 1.

**Table 1. Characteristics of the stimuli before running the matching procedure (standard deviations are shown in parentheses)**

Condition	FRE	LNG	SYL	N	NHF	BFQ	IMA
(1) Low-frequency	7.74 (3.82)	5.84 (1.66)	1.87 (0.80)	2.76 (3.89)	1.30 (2.39)	889.46 (755.19)	579.77 (66.21)
(2) Medium-frequency	34.31 (13.31)	5.33 (1.51)	1.60 (0.67)	4.08 (5.27)	0.93 (1.89)	1178.15 (1015.28)	564.15 (83.73)
(3) High-frequency	176.91 (169.17)	4.61 (1.18)	1.34 (0.54)	5.70 (5.44)	0.70 (1.05)	1620.50 (1382.17)	577.80 (65.43)

*Note.* FRE = word frequency per million; LNG = word length; SYL = number of syllables; N = number of neighbors; NHF = number of higher frequency neighbors; BFQ = mean bigram frequency; IMA = imageability.

In this initial set, conditions differ in frequency,  $F(2, 199) = 56.68$ ,  $p < .001$ , word length,  $F(2, 199) = 12.46$ ,  $p < .001$ , number of syllables,  $F(2, 199) = 10.70$ ,  $p < .001$ , number of neighbors,  $F(2, 199) = 6.34$ ,  $p < .005$ , and mean bigram frequency,  $F(2, 199) = 8.05$ ,  $p < .001$ . On the other hand, number of higher frequency neighbors,  $F(2, 199) = 1.85$ ,  $p = .16$ , and imageability ratings,  $F(2, 199) = 0.89$ ,  $p = .41$ , are similar across conditions.

**Step 2** consisted in the normalization of the variables to be matched in order to get an optimal matching solution. There are two ways to convert values to z-scores using SPSS. The first one is through the visual interface, where we have to click on “Analyze”, “Descriptive Statistics”, and then

“Descriptives”. After selecting all the variables to be normalized, we have to check the box labeled as “Save standardized values as variables” and then accept the selection. After that, SPSS will create in the active dataset the new z-scored variables with the same name as the old variables, but preceded by a capitalized z (e.g., “var1” is converted to “Zvar1”). The other method to normalize variables is to use the following SPSS syntax:

```
DESCRIPTIVES VARIABLES = var1 var2 var3
/SAVE.
```

In the above syntax, “var1”, “var2”, “var3”, etc. represent the names of the variables to be normalized. In our example, z-scored variables to be matched were "ZLNG", "ZSYL", "ZN", "ZNHF", "ZBFQ", and "ZIMA" (see Table 1 for a description).

**Step 3** is a crucial one, as it will guide us to an appropriate matching solution. This step begins with the estimation of *k-max* (**Step 3.1**), which is equal to the number of stimuli in the condition with fewer stimuli. In our example, *k-max* corresponded to 60, that is, the number of stimuli in the medium-frequency condition.

Then, **Steps 3.2** and **3.3** involve examining the number of valid items and the sum of their distances to the center of the clusters across the possible *k* values. As stated before, computing these indices is not trivial, as it involves many runs of a cluster analysis. For this reason, we have devised a SPSS macro to do it, available as a supplementary material to this paper. Previously to using the SPSS macro, users have to tweak some values to adjust it to their datasets. All the values that need a setting are located in line 100:

```
genClust maxClusters=60 condVar=COND firstCond=1 lastCond=3
variablesToMatch = ZLNG ZSYL ZN ZNHF ZBFQ ZIMA.
```

In this line, “maxClusters” is the value of *k-max* (i.e., 60 in our example). The variable “condVar” is the name of the variable in the SPSS dataset that contains the discretization of the critical variables through conditions (i.e., COND in our example). The variables “firstCond” and “lastCond” contain respectively the number of the first condition (i.e., 1: low-frequency words), and the number of the last condition (i.e., 3: high-frequency words). Finally, the “variablesToMatch” parameter is a space-delimited list of the names of all the variables— previously normalized— that need to be matched across conditions (i.e.: ZLNG, ZSYL, ZN, ZNHF, ZBFQ, and ZIMA in our example).

Now, we can already execute the SPSS macro. Note that it will work only with SPSS version 22 or higher. The script will run up to *k-max* cluster

analyses saving in a new dataset the relevant information from the outcome of each analysis. The newly created dataset will contain as many rows as analyses performed, each including the following information (see Figure 2):

1) *numClusters*: number of clusters selected for running the analysis (sorted from two to *k-max*).

2) *validClusters*: number of valid clusters, that is, clusters having at least one matched stimulus of each condition.

3) *stimuliPerCondition*: number of matched stimuli per condition that can be obtained with the selected number of clusters.

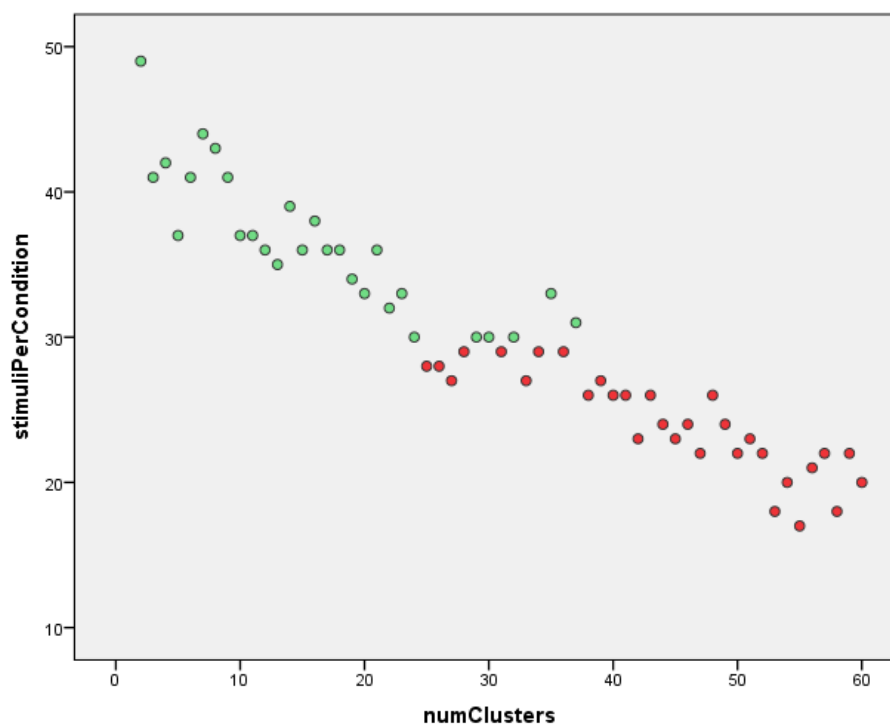
4) *sumOfDistances*: sum of distances from each stimulus to the center of its cluster.

	numClusters	validClusters	stimuliPerCondition	sumOfDistances
1	2	2	49	790
2	3	3	41	635
3	4	4	42	544
4	5	5	37	481
5	6	6	41	455
6	7	6	44	394
7	8	7	43	360
8	9	7	41	362
9	10	6	37	343
10	11	5	37	317

**Figure 2. Extract from the dataset created after executing the SPSS macro.**

Additionally, after the execution of the macro, the SPSS results window will show two plots. The first plot shows, for each cluster analysis performed, the number of matched stimuli per condition. Since in our example we wanted to obtain 30 stimuli per condition, only those analyses resulting in 30 or more than 30 matched stimuli will be taken into consideration (see Figure 3).

The second plot displays the sum of distances of each item to its cluster center, for each of the analyses run. In our example (see Figure 4), and as it could be expected, the sum of distances decreased as the number of clusters increased—as clusters become smaller, they also become tighter.

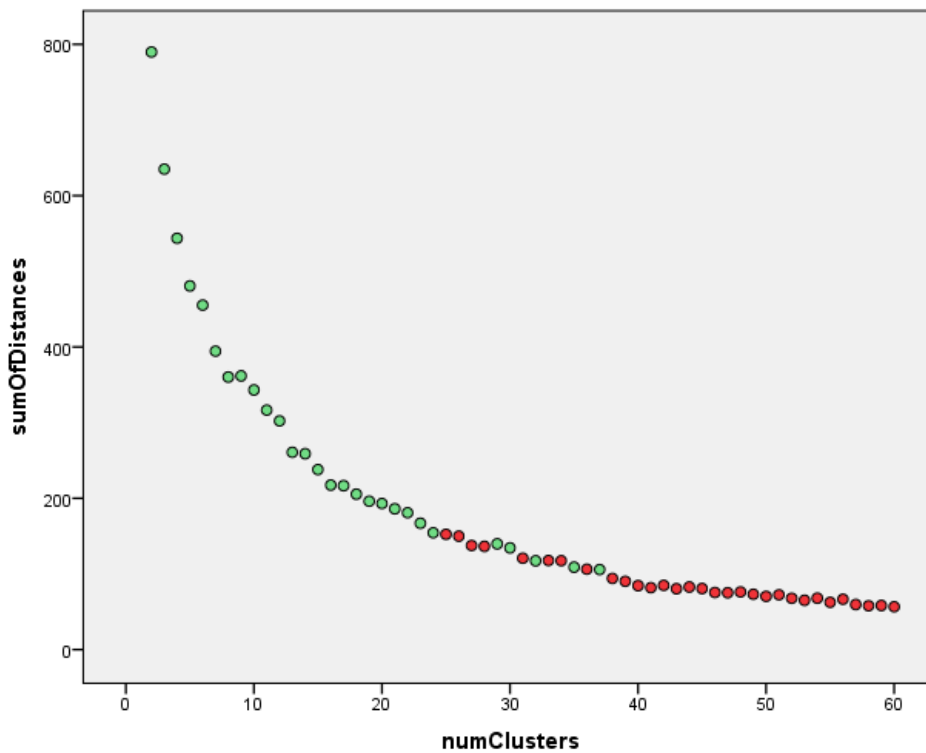


**Figure 3. Number of matched stimuli per condition for each cluster analysis. Cluster analyses with 30 or more matched stimuli per condition are colored in green.**

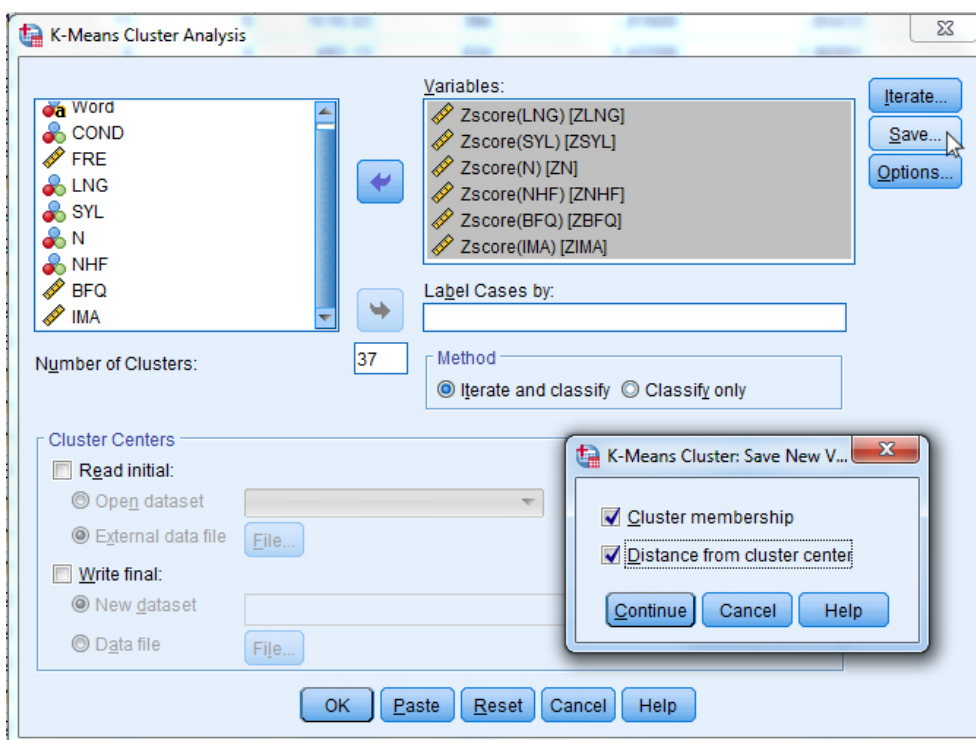
All the above information is decisive for selecting the appropriate  $k$  value. This information can be observed graphically, or can be studied numerically from the new dataset created by the SPSS macro. In general, a good matching solution will be that providing the desired number of matched stimuli (i.e., *stimuliPerCondition*), while having a low sum of distances (i.e., *sumOfDistances*). Thus, one good matching solution for our example is that corresponding to a classification in 37 clusters. This clustering would provide 31 matched stimuli per condition, while having the lowest sum of distances among the valid solutions.

In the next step (**Step 4**), we carried out the K-means cluster analysis with the obtained  $k$  value of 37. This step can be achieved easily using the visual interface, going through the option “Analyze”, “Classify”, and then “K-means cluster”. In the dialog box, the user only has to select the variables to match (after normalization), to indicate the desired number of clusters, and to mark both check boxes under the save button: “cluster

membership”, and “distance from cluster center” (see Figure 5). The “cluster membership” check box will create a new variable in the dataset named “QCL\_1”, that indicates to which cluster belongs each stimulus. The second check box (i.e., “distance from cluster center”) will create a second variable named “QCL\_2” that shows the Euclidean distance from the stimulus to the center of its cluster. In the following steps, we will refer to these variables as “cluster\_number”, and “cluster\_distance”, respectively. To perform the following steps, it is necessary to sort the data first by cluster number, and then by condition.



**Figure 4.** Sum of distances of each item to its cluster center for each cluster analysis. Cases colored in green correspond to those with 30 or more matched stimuli per condition.



**Figure 5.** SPSS dialog box for K-Means Cluster Analysis with the Save button sub-dialog.

**Step 5** consisted in selecting the final items from the raw solution. First, we removed those clusters with missing stimuli in at least one of the experimental conditions. For example, examine cluster 10 in Figure 6:

	Word	COND	cluster_number	cluster_distance
46	prison	3	9	,54545
47	bold	1	10	,72601
48	tight	2	10	,72601
49	gas	3	11	,00000

**Figure 6.** Screenshot of the SPSS dataset depicting cluster 10.

Cluster 10 includes stimuli from conditions 1 and 2, but not from condition 3. Thus, it should be removed from the dataset. After removing all the invalid clusters, we should revise those clusters having different number of stimuli per condition (e.g., see Figure 7):

	Word	COND	cluster_number	cluster_distance
180	game	3	33	,77452
181	maze	1	34	,65499
182	nail	1	34	,32222
183	spice	1	34	,92273
184	worm	1	34	,77556
185	gate	2	34	1,16763
186	sex	3	34	1,29332
187	mediocre	1	35	,00000

**Figure 7. Screenshot of the SPSS dataset depicting cluster 34.**

For example, suppose we have a cluster as number 34. This cluster contains four stimuli from condition 1, but only one from conditions 2 and 3. Thus, in order to match the number of stimuli per condition in the cluster, we must delete three stimuli from condition 1. The strategy adopted here was to keep the stimuli from condition 1 whose distance to the cluster center was more similar to the distances of the stimuli from conditions 2 and 3. In this case, we kept the stimulus “spice”, as it is the stimulus from condition 1 with the most similar distance to “gate” (from condition 2), and “sex” (from condition 3).

Once finished the trimming process, we will have the desired number of stimuli per condition, or even more. In this example, we aimed to have 30 stimuli per condition, although we selected a  $k$  value that gave as a result 31 items per condition. Thus, it was necessary to remove three stimuli, one from each condition, with the requirement that they had to belong to the same cluster. This pruning can be done following the experimenter’s own intuitions. However, to avoid biases, it is desirable to follow a more objective procedure. For instance, in our example we opted for removing one item from each condition in cluster 9, trying to equate as much as possible the cluster distances between conditions (see Figure 8). Thus, we kept those stimuli with the most similar distances, that is: fungus (0.57)



from condition 1, stomach (0.55) from condition 2, and prison (0.55) from condition 3.

	Word	COND	cluster_number	cluster_distance
19	family	3	4	,89874
20	fungus	1	9	,56890
21	magnet	1	9	,52659
22	passport	2	9	,90699
23	stomach	2	9	,55437
24	captain	3	9	,84834
25	prison	3	9	,54545
26	carrot	1	13	,61019

**Figure 8.** Screenshot of the SPSS dataset depicting cluster 9 after matching the number of stimuli per condition in a cluster.

Finally, we obtained 30 stimuli per condition and we were ready to perform the last step (**Step 6**): to assess the suitability of the solution obtained. The properties of the selected words by condition are shown in Table 2.

After running the entire procedure, all the variables to be controlled were successfully matched among conditions: word length,  $F(2, 89) = 0.14$ ,  $p = .87$ ; number of syllables,  $F(2, 89) = 0.03$ ,  $p = .97$ ; number of neighbors,  $F(2, 89) = 1.04$ ,  $p = .36$ ; number of higher frequency neighbors,  $F(2, 89) = 0.29$ ,  $p = .75$ ; mean bigram frequency,  $F(2, 89) = 0.41$ ,  $p = .66$ ; and imageability,  $F(2, 89) = 0.10$ ,  $p = .90$ . Additionally, we ran independent sample t-tests between each of the experimental conditions and the other experimental conditions, for the six variables to be controlled. None of the analyses reached significance (all  $ps > .23$ ). Of course, word frequency remained different among conditions,  $F(2, 89) = 53.16$ ,  $p < .001$ . Low-frequency words had a mean frequency of 8.00 ( $SD = 3.89$ ), medium-frequency words had a mean value of 32.25 ( $SD = 14.26$ ), and high-frequency words had a mean frequency of 152.98 ( $SD = 99.95$ ).

**Table 2. Characteristics of the stimuli after running the matching procedure (standard deviations are shown in parentheses)**

Condition	FRE	LNG	SYL	N	NHF	BFQ	IMA
(1) Low-frequency	8.00 (3.89)	5.37 (1.13)	1.60 (0.62)	1.87 (2.10)	0.63 (1.27)	907.01 (802.48)	593.43 (44.95)
(2) Medium-frequency	32.26 (14.26)	5.23 (1.14)	1.63 (0.61)	2.70 (3.14)	0.50 (1.28)	931.68 (594.77)	594.63 (48.01)
(3) High-frequency	152.98 (99.95)	5.23 (1.07)	1.60 (0.62)	1.93 (2.08)	0.40 (0.97)	1060.17 (691.61)	589.47 (45.72)

*Note.* FRE = word frequency per million; LNG = word length; SYL = number of syllables; N = number of neighbors; NHF = number of higher frequency neighbors; BFQ = mean bigram frequency; IMA = imageability.

### Conclusion

The algorithm described here can be very useful for psycholinguists in selecting well-controlled materials for experimental designs. They will save time while committing fewer errors. Although other methods have been proposed to achieve the same aim, as Match (Van Casteren & Davis, 2007) and SOS (Armstrong et al., 2012), we think that our algorithm is complementary to them rather than redundant. Match and SOS provide a closed solution. They provide a final set of matching stimuli, where the user has no insight on the intermediate states followed to achieve it. In contrast, the clustering method proposed here is open. By following the procedure step by step, experimenters become familiar with their materials. This fact benefits the assessment of the generalizability of the selected stimuli set. Indeed, if there is a low number of valid solutions when assessing the optimal  $k$ -value, the effects found with our set of items will probably not be generalizable to other stimuli sets.

Furthermore, the method presented here does not provide just a solution, but a range of possible solutions that the user can tweak to reach a satisfactory set of materials. It has been argued (Van Casteren & Davis, 2007), that a fully automated procedure to select stimuli is desirable to avoid the experimenter bias described by Forster (2000). According to this

bias, it is possible that experimenters—consciously or not—choose those items that favor their hypothesis. Although our procedure is not completely blind for the experimenter, the algorithm-form of the procedure protects against this bias, as the item selection criteria are narrow enough to avoid it. Only steps 5.2 and 5.3 (the surplus-removing steps) leave room for the influence of the experimenter bias. However, this can be avoided by specifying a strict selection criterion in advance. On the other hand, this fact may become an advantage under some circumstances. For instance, suppose that two similar words (e.g., two synonyms or the same word in both genders) end up in the same experimental condition, which may hinder the generalizability of the stimuli set. With a closed procedure, this fact would go unnoticed until the end of the process, whereas with our algorithm it can be proactively detected.

In sum, the main aim of the present work was to describe a fast and simple method to match words for psycholinguistic experiments. Moreover, with our algorithm we hope that Cutler's (1981) prediction of being lost for words will be postponed for another decade.

## RESUMEN

**Agrupar palabras para igualar condiciones: Un algoritmo para la selección de palabras en diseños factoriales.** Con el creciente refinamiento de los modelos de procesamiento del lenguaje y los nuevos hallazgos sobre qué variables pueden modular dichos procesos, la selección de palabras para experimentos de diseño factorial se está convirtiendo en una tarea cada vez más ardua. Seleccionar conjuntos de palabras que difieren en una variable pero que están igualadas en una decena de posibles variables extrañas, lleva mucho tiempo y está sujeto a errores. Para ayudar a los experimentadores en esta desagradecida tarea, presentamos un método sencillo que permite realizarla con poco esfuerzo. El método se basa en el agrupamiento de K-medias para identificar conjuntos pequeños y compactos de palabras igualadas en las variables deseadas. El procedimiento ha sido formalizado en un algoritmo, esto es, una serie de pasos concretos y sencillos de seguir. Además, también aportamos la sintaxis en SPSS para ayudar en la selección del número adecuado de agrupaciones. Tras una revisión de la teoría, presentamos un ejemplo práctico que guiará al lector a través del procedimiento completo. El conjunto de datos del ejemplo se encuentra disponible como material complementario a este artículo.

## REFERENCES

- Armstrong, B. C., Watson, C. E., & Plaut, D. C. (2012). SOS! An algorithm and software for the stochastic optimization of stimuli. *Behavior Research Methods*, *44*, 675–705. doi: 10.3758/s13428-011-0182-9
- Baayen, R. H. (2004). Statistics in psycholinguistics: A critique of some current gold standards. *Mental Lexicon Working Papers*, *1*, 1–45.
- Baayen, R. H. (2010). A real experiment is a factorial experiment? *Mental Lexicon*, *5*, 149–157. doi: 10.1075/ml.5.1.06baa
- Cohen, J. (1983). The cost of dichotomization. *Applied Psychological Measurement*, *7*, 249–253. doi: 10.1177/014662168300700301
- Coltheart, M. (1981). The MRC psycholinguistic database. *The Quarterly Journal of Experimental Psychology*, *33*, 497–505. doi: 10.1080/14640748108400805
- Cutler, A. (1981). Making up materials is a confounded nuisance, or: Will we be able to run any psycholinguistic experiments at all in 1990? *Cognition*, *10*, 65–70. doi: 10.1016/0010-0277(81)90026-3
- Davis, C. J. (2005). N-Watch: A program for deriving neighborhood size and other psycholinguistic statistics. *Behavior Research Methods*, *37*, 65–70. doi: 10.3758/BF03206399
- Davis, C. J., & Perea, M. (2005). BuscaPalabras: A program for deriving orthographic and phonological neighborhood statistics and other psycholinguistic indices in Spanish. *Behavior Research Methods*, *37*, 665–671. doi: 10.3758/BF03192738
- Díez, E., Fernández, A., & Alonso, M. A. (2006). *NIPE: Normas e índices de interés en Psicología Experimental*. Retrieved from <http://campus.usal.es/~gimc/nipe/>
- Duchon, A., Perea, M., Sebastián-Gallés, N., Martí, A., & Carreiras, M. (2013). EsPal: One-stop shopping for Spanish word properties. *Behavior Research Methods*, *45*, 1246–1258. doi: 10.3758/s13428-013-0326-1
- Forster, K. I. (2000). The potential for experimenter bias effects in word recognition experiments. *Memory & Cognition*, *28*, 1109–1115. doi: 10.3758/BF03211812
- Guasch, M., Boada, R., Ferré, P., & Sánchez-Casas, R. (2013). NIM: A Web-based Swiss Army knife to select stimuli for psycholinguistic studies. *Behavior Research Methods*, *45*, 765–771. doi: 10.3758/s13428-012-0296-8
- Hillhouse, J. J., & Adler, C. M. (1997). Investigating stress effect patterns in hospital staff nurses: results of a cluster analysis. *Social Science & Medicine*, *45*, 1781–1788. doi: 10.1016/S0277-9536(97)00109-3
- Jain, A. K. (2010). Data clustering: 50 years beyond K-means. *Pattern Recognition Letters*, *31*, 651–666. doi: 10.1016/j.patrec.2009.09.011
- Lorr, M., & Strack, S. (1994). Personality profiles of police candidates. *Journal of Clinical Psychology*, *50*, 200–207. doi: 10.1002/1097-4679(199403)50:2<200::AID-JCLP2270500208>3.0.CO;2-1
- Rokach, L., & Maimon, O. (2005). Clustering methods. In O. Maimon & L. Rokach (Eds.), *The data mining and knowledge discovery handbook* (pp. 321–352). Boston, MA: Springer US. doi: 10.1007/0-387-25465-X\_15
- Sparks, R. L., Patton, J., & Ganschow, L. (2012). Profiles of more and less successful L2 learners: A cluster analysis study. *Learning and Individual Differences*, *22*, 463–472. doi: 10.1016/j.lindif.2012.03.009
- Troche, J., Crutch, S., & Reilly, J. (2014). Clustering, hierarchical organization, and the topography of abstract and concrete nouns. *Frontiers in Psychology*, *5*. doi: 10.3389/fpsyg.2014.00360

Van Casteren, M., & Davis, M. H. (2007). Match: A program to assist in matching the conditions of factorial experiments. *Behavior Research Methods*, 39, 973–978. doi: 10.3758/BF03192992

(Manuscript received: 14 April 2016; accepted: 21 June 2016)